Nonparametric Bayesian Models for Joint Analysis of Imagery and Text

by

Lingbo Li

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____
Lawrence Carin, Supervisor

_____
Robert Calderbank

_____
Guillermo Sapiro

_____
David Dunson

_____
Mauro Maggioni

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering in the Graduate School of Duke University
2014

Abstract

Nonparametric Bayesian Models for Joint Analysis of Imagery and Text

by

Lingbo Li

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____
Lawrence Carin, Supervisor

_____
Robert Calderbank

_____
Guillermo Sapiro

_____
David Dunson

_____
Mauro Maggioni

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering
in the Graduate School of Duke University
2014

Abstract

It has been increasingly important to develop statistical models to manage large-scale high-dimensional image data. This thesis presents novel hierarchical nonparametric Bayesian models for joint analysis of imagery and text. This thesis consists two main parts.

The first part is based on single image processing. We first present a spatially dependent model for simultaneous image segmentation and interpretation. Given a corrupted image, by imposing spatial inter-relationships within imagery, the model not only improves reconstruction performance but also yields smooth segmentation. Then we develop online variational Bayesian algorithm for dictionary learning to process large-scale datasets, based on online stochastic optimization with a natural gradient step. We show that dictionary is learned simultaneously with image reconstruction on large natural images containing tens of millions of pixels.

The second part applies dictionary learning for joint analysis of multiple images and text to infer relationship among images. We show that feature extraction and image organization with annotation (when available) can be integrated by unifying dictionary learning and hierarchical topic modeling. We present image organization in both "flat" and hierarchical constructions. Compared with traditional algorithms feature extraction is separated from model learning, our algorithms not only better fits the datasets, but also provides richer and more interpretable structures of images.

Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

BP	beta process
BeP	Bernoulli process
BPFA	beta process factor analysis
BNP	bayesian nonparametric
CRP	Chinese restaurant process
CS	compressive sensing
DCT	discrete cosine transform
DDP	dependent Dirichlet process
DP	Dirichlet process
HBP	hierarchical beta process
HDP	hierarchical Dirichlet process
LDA	latent Dirichlet allocation
KNN	k-nearest neighbor
KSBP	kernel stick-breaking process
nCRP	nested Chinese restaurant process
nDP	nested Dirichlet process
PCA	principal component analysis
PSBP	Probit stick-breaking process
PSNR	peak signal-to-noise ratio
SIFT	scale-invariant feature transform

SVD	singular value decomposition
VB	variational Bayesian
VQ	vector quantization

Acknowledgements

First of all, I feel truly grateful to my advisor, Dr. Lawrence Carin, for his systematic guidance, inspiration, earnest encouragement during my past years of study. Without his tremendous support, strict requirements and vast knowledge, this work would not be finished. His rigorous and diligent working attitude will always influence me through the whole life.

Thanks to my other committee members including Dr. Robert Calderbank, Dr. David Dunson, Dr. Guillermo Sapiro, and Dr. Mauro Maggioni for their time and comments on this thesis.

I would like to give thanks to all my former and current group members. Especially I would like to express my gratitudes to Dr. Lu Ren for her guidance and tremendous help both in group and later during my intern at Apple. I would like to thank Dr. Lihan He for his guidance in the beginning of my PhD study. Thanks to Dr. Xuejun Liao for his valuable discussions and suggestions on my research. Besides, I want to thank other colleagues in my group, including Dr. Bo Chen, Dr. Esther Salazar, Dr. Chunping Wang, Dr. Jorge Silva, Dr. Haojun Chen, Dr. Lan Du, Dr. Minhua Chen, Dr. Eric Wang, Dr. Qi An, Zhengming Xing, Xianxing Zhang, Miao Liu, Shaobo Han, David Carlson, Wenzhao Lian, for their collaboration, assistance and valuable discussions. I feel so lucky to work in such a wonderful group with so many smart and great people.

I would like to thank Dr. John Hershey and Dr. Belle Tseng for providing

me invaluable opportunities of internship in Mitsubishi Electrical Research Lab and Apple Inc respectively. Especially to Belle, thanks to her for providing me extreme support to both of my work and life.

Finally, and most importantly, I would like to thank Dr. Mingyuan Zhou who has always been standing by me with his endless support, encouragement and love for years. I would like to thank my dear parents for raising me up in a happy family. Their support and love are always the heart of my harbor. This dissertation is dedicated to them.

1

Introduction

In the current era, machine learning has become an essential technique for researchers to understand and explore hidden information from the data. For example, simultaneous image reconstruction and segmentation for a corrupted image can be used in medical imaging, meteorological analysis and sensing applications. Large-scale latent structure discovery of high-dimensional data can be used to reconstruct massive incomplete datasets. Automatically organizing imagery and associated text can be used to improve multimedia library indexing, retrieval and organization.

With the exponential growth of image data, a critical question for researchers is how to accurately and insightfully reveal images for humans. Visual signals are complex and variable, which makes them challenging to semantically understand. This thesis develops rigorous machine learning algorithms for imagery analysis with text information (when available), especially focusing on image segmentation and organization. We will present several novel nonparametric Bayesian models at multiple depths: large-scale corrupted image reconstruction with segmentation, image organization with feature learning, and hierarchical image structure learning.

Bayesian statistics is an important approach in machine learning via updating

our beliefs about the form, structure or the distribution of the data based on the observations. The challenging part is how to update our beliefs to learn the appropriate model parameters. Often, data is high-dimensional and massive, which requires richer models and more parameters to regularize. Incorporating Bayesian nonparametric priors into models allows the data to be automatically modeled without defining its complexity *a priori*, which enables us to define our beliefs over an infinite dimensional parameter space.

One typical example using nonparametric Bayesian priors is to cluster data into groups. Given thousands of natural images collected from flickr, one may be interested: How do we set the number of groups for these images? The traditional approach to this *unsupervised clustering* question is via cross-validation, which may lead to model overfitting or underfitting problems. Alternatively recent researchers develop the nonparametric Bayesian approach of Gaussian mixture models. Given a collection of data, Gaussian mixture models assume each data point to be generated by a mixture of hidden components, each represented by a Gaussian distribution. The number of Gaussians to be chosen can be automatically inferred via the Dirichlet process (DP) (Antoniak, 1974; E.Rasmussen, 2000; MacEachern and Muller, 1998) prior. The DP prior is usually imposed on the countably infinite component space to allow the model itself to automatically determine the appropriate number of mixture components. Mixture modeling has become one of the most useful tools in statistics, machine learning and data mining for applications involving density estimation or clustering.

When modeling an image, the collection of all possibly overlapping patches is often used to represent it. Thus image segmentation could be achieved by inferring latent component index for each pixel (representing the associated patch) via clustering pixel values with DP mixture models. The components in mixture models are exchangeable, which may not be appropriate for some scenarios. For an image with

blue sky on top, a pond at bottom and lawn in between, the sky and the pond may be much likely to be clustered into the same segment, since their pixel values are similar as both in blue. To alleviate this issue, constraint such as dependence is usually imposed upon the associate weights of components via extra information appended with data. For this case, spatial dependence is imposed by assuming pixels physically nearby should share similar components besides their pixel values. By incorporating spatial locations of pixels, it is unlikely to cluster the sky and the pond region into one segment since they are physically separated apart in the image. Nonparametric Bayesian methods have been successfully applied in such field (MacEachern, 1999) with applications of processing audio signals (Ren et al., 2009), spatial locations (Duan et al., 2007a; Dunson and Park, 2007; Ren et al., 2011), and time evolving data (Blei and Lafferty, 2006).

Unlike traditional approaches that extract various types of features from image patches in advance, we will develop *factorial* models so as to avoid defining which features to use and selecting the size of feature codebook at the preprocessing step. Instead of assuming each patch to be associated with one single component in mixture models, factorial models allow each patch to be described by multiple components. For K disjoint components, there are 2^K different configurations, which makes it challenging to determine how many and which components to be selected for each data. To tackle this challenge, another nonparametric Bayesian prior, *beta process* (BP) is proposed that each observation is represented by a subset of latent components, the expected number of which growing with the size of data. This prior has been mostly used in a *feature model* with an unbounded number of potential features. Given a collection of data, each one is represented by multiple features, the entire set of which will be shared across all observations. To model high-dimensional data like image patches, sparseness is often imposed in the feature space to restrict the model to be succinct and identifiable. Sparseness means that each data can be

linearly represented by only a few features so that the weights of many other features are zero or close to zero. By incorporating this BP prior, the whole learning process including how many and which features to be chosen could be automatically inferred from the observations. This also refers to the *dictionary learning* model, which has been successfully used in many signal and image processing (Mairal et al., 2010a, 2009b, 2008d; Zhou et al., 2011a).

Leveraging the advantage of nonparametric Bayesian modeling that allows many exotic stochastic process as priors, for single image, dictionary learning could be combined with segmentation modeling to achieve image reconstruction and segmentation simultaneously as elaborated in Chapter 2. For multiple images, dictionary learning could also be integrated with hierarchical modeling to infer relationship among large volume of images. Motivated by hierarchical Dirichlet process (HDP) (Teh et al., 2004b), one upper "category" level is imposed on top of the latent components in mixture models to allow each category sharing components with different probabilities. The visual modeling could infer the category index to each image, the latent component index to each patch, and which features to represent each patch. Furthermore, text modeling could be incorporated together with the visual modeling to guide the meaningful organization of images.

Followed by the "flat" organization of images introduced in Chapter 4, a hierarchy could further be explored for describing relationship among images as descried in Chapter 5. Recently researchers develop nonparametric Bayesian hierarchical structured representations of data mainly for text analysis (Blei et al., 2010; Teh et al., 2004b). By incorporating both visual and word information, a semantically and visually meaningful hierarchical tree-based dictionary learning structure model is presented for jointly modeling images and the associated text counterparts (when available). Nonparametric Bayesian priors encourage this hierarchical tree structure to grow in complexity with the data by inferring the width and depth of the tree.

This hierarchical structure could not only better help researchers to organize images and texts, but provides a more accurate and meaningful solution to image retrieval, annotation and classification.

Besides the model representation side, researchers have become more and more interested in how to develop efficient algorithms to manage large-scale datasets. Due to the limits of memory and computational problems, traditional batch algorithms are infeasible to process the real-world datasets, such as online images from flickr or an image with tens of millions of pixels. Recent work (Hoffman et al., 2010; Wang et al., 2011) of online learning algorithms have been successfully implemented for large-scale text analysis. For imagery, we will extend our nonparametric Bayesian batch-learning algorithm into both online learning and parallel paradigm to perform dictionary learning and image reconstruction simultaneously for a very large image with tens of millions of pixels as presented in Chapter 3.

The rest of this chapter is organized as follows: Section 1.1 provides background material on the stochastic processes in developing Bayesian nonparametric models that we use: the Dirichlet process in two representations, its hierarchical extension, and the beta process. Section 1.2 briefly reviews Bayesian Dictionary Learning (Paisley and Carin, 2009a; Zhou et al., 2009) with beta process factor analysis (BPFA) model for recovery of imagery based upon compressive, incomplete and/or noisy measurements. This BPFA model allows for each observation to be sparsely represented by a set of factors that are shared among all observations. Specifically a truncated beta-Bernoulli process is employed to infer an appropriate dictionary for the observations under test, and also for image recovery. The non-parametric method also allows the noise variance to be unknown and non-stationary. Section 1.3 first summarizes the latent Dirichlet allocation (LDA) model and its hierarchical extension in modeling text corpus, followed by reviewing how such topic models are applied in imagery analysis. The organization of this thesis is provided in Section 1.4.

1.1 Nonparametric Bayesian Methods

In Bayesian statistics, nonparametric models are constructed through priors on function spaces of distributions whose class should be analytically tractable and rich in the sense of having a large enough support. Therefore Bayesian nonparametric models are not parameter-free, but have an infinite number of parameters. If suitably designed, these methods could have efficient posterior inference. In the following sections, we briefly describe some classes of Bayesian nonparametric methods: the Dirichlet process, its hierarchical extension, and the beta process.

1.1.1 Dirichlet Processes

The Dirichlet process (DP) (Ferguson, 1973b) constitutes a popular means of performing nonparametric clustering. A random draw from a DP is denoted as $G \sim \mathrm{DP}(\alpha G_0)$, with precision $\alpha \in \mathbb{R}^+$ and "base" measure G_0. G_0 can be any continuous or discrete distribution over Θ. G is defined as Dirichlet Process if for any random finite measurable partition $\Omega_1, \Omega_2, ..., \Omega_K$ of Θ, the vector $(G(\Omega_1), G(\Omega_2), ..., G(\Omega_K))$ is of Dirichlet distributed as

$$(G(\Omega_1), G(\Omega_2), ..., G(\Omega_K)) \sim \mathrm{Dir}(\alpha G_0(\Omega_1), ..., \alpha G_0(\Omega_K)) \tag{1.1}$$

Based on the above definition, Dirichlet Distribution has the following properties: The base distribution is the mean of the DP, i.e., for any measurable partitions $\mathbf{\Omega}$, we have $\mathbb{E}[G(\mathbf{\Omega})] = G_0(\mathbf{\Omega})$. Additionally, $\mathrm{var}[G(\mathbf{\Omega})] = \frac{G_0(\mathbf{\Omega})(1-G_0(\mathbf{\Omega}))}{\alpha+1}$ indicates that the concentration parameter α inversely controls how $G(\mathbf{\Omega})$ is different from its mean $G_0(\mathbf{\Omega})$. As $\alpha \to \infty$, $G(\mathbf{\Omega}) \to G_0(\mathbf{\Omega})$ for any measurable $\mathbf{\Omega}$.

There are two main constructions to draw samples from a DP. One is the Chinese restaurant process (CRP) through P*ó*lya urn scheme (Aldous, 1985; Blackwell and MacQueen, 1973), and the other one is the stick-breaking construction (Sethuraman, 1994a).

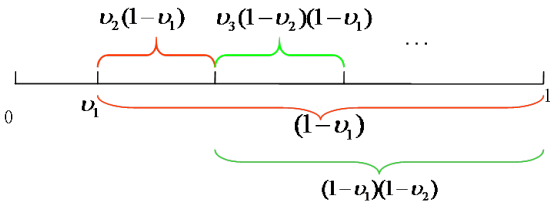


FIGURE 1.1: An example of stick-breaking process.

Chinese Restaurant Process

The Chinese restaurant process is a random process with infinitely many tables $\{\phi_k\}_{k=1}^{\infty}$, each of which can seat infinite customers. The general process of CRP is stated as follows: The first customer enters the restaurant and sits at the first table. The second customer enters and decides either to sit with the first customer, or by herself at a new table. In general, the n+1st customer either joins an already occupied table k with probability proportional to the number N_k of customers already sitting there, or sits at a new table with probability proportional to α. Let $\{\phi_k\}_{k=1}^{\infty}$ denote the unique value of $\{\theta_n\}_{n=1}^{N}$. By integrating out G, the conditional distribution of θ_N given $\theta_1, \ldots, \theta_{N-1}$ is

$$\theta_N|\theta_1, ..., \theta_{N-1}, \alpha, G_0 \sim \sum_{k=1}^{K} \frac{N_k}{n-1+\alpha}\delta_{\phi_k} + \frac{\alpha}{n-1+\alpha}G_0 \tag{1.2}$$

The number of tables could be infinite with new customers coming in. Notice that α controls the number of clusters in a direct manner, with larger α implying a larger number of clusters a priori. This intuition will help in the application of DPs to mixture models.

Stick-Breaking Construction

Another perspective of DP is the stick-breaking construction (Sethuraman, 1994a) as

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}, \quad \pi_k = \nu_k \prod_{k'=1}^{k-1} \nu_{k'}, \quad \nu_k \overset{\text{iid}}{\sim} \text{Beta}(1, \alpha), \quad \phi_k \overset{\text{iid}}{\sim} G_0, k = 1, \ldots, \infty \tag{1.3}$$

The $\{\pi_k\}_{k=1}^{\infty}$ may be viewed as a sequence of fractional breaks from a "stick" of original length one, where the fraction of stick broken off on break k is ν_k. The ϕ_k are model parameters, associated with the kth data cluster. α indirectly controls the number of clusters. If α is large, the fraction of stick broken off drawn from Beta$(1, \alpha)$ becomes small. Considering the total length to be one, there are more sticks broken off each with relatively small length, which generates more clusters. In contrast, fewer longer sticks are broken off if α is small, corresponding to fewer clusters generated. One remarkable property of this stick-breaking construction is the weights of all sticks/draws sums to 1 as $\sum_{k=1}^{\infty} \pi_k = 1$, which naturally makes the DP suitable for the mixture modeling problem. This stick-breaking representation has been exploited to be mathematically efficient on the MCMC inference, due to which many generalizations that allow for dependence across a collection of distributions have been built upon it, including the nCRP(Blei et al., 2010), the DDP (MacEachern, 1999), the PSBP (Rodriguez and Dunson, 2011), the LSBP (Ren et al., 2011).

Dirichlet Process Mixture Models

Due to the property that the probabilities of all draws from the DP sums to 1, DP usually serves as a prior over potentially infinite number of latent components in a mixture model. This DP mixture model generates data $\{x_i\}_{i=1}^{N}$ as follows:

$$x_i|\theta_i \sim F(\theta_i), \quad \theta_i|G \sim G, \quad G|\alpha, G_0 \sim \mathrm{DP}(\alpha G_0) \tag{1.4}$$

where $F(\theta_i)$ is the likelihood of data x_i given θ_i, which is conditionally independent given G. Again x_i is conditionally independent given θ_i. Since more than one θ_i's could take the same value, those data that share the same parameter θ belong to one cluster. To simplify the inference, an indicator variable z_i for each data is introduced described as

$$x_i|\{\theta_k\}_{k=1}^{\infty}, z_i \sim F(\theta_{z_i}), \quad \theta_k|G \sim G, \quad z_i|\boldsymbol{\pi} \sim \boldsymbol{\pi} \tag{1.5}$$

where $\boldsymbol{\pi}$ could be constructed as 1.3. Therefore both Gibbs sampling (Neal, 2000)

and variational inference (Blei and Jordan, 2004) could be easily derived for the approximate posterior inference for this DP mixture models.

1.1.2 Hierarchical Dirichlet Process

The Dirichlet Process has been widely used for data clustering problems mainly within one group. There are some scenarios that data need to be partitioned into multiple groups, each containing shared mixture components with different proportion. Take organizing images for example. Images are often represented as "bags of words". Patches of each image are clustered into many components named as "objects", and these "objects" will be shared across all other images from other groups. Different groups of images may exhibit different proportions over "objects". The hierarchical Dirichlet Process (HDP) extends the DP to such scenario as a hierarchical generalization of the DP. Each group G_j can be represented as a DP mixture model associated with a draw from a group-specific Dirichlet process

$$G_j|G_0, \alpha \sim \mathcal{DP}(\alpha G_0) \tag{1.6}$$

In order to allow for sharing of components across different groups, the base measure G_0 is required to be discrete that G_0 itself is drawn from an upper-level Dirichlet process

$$G_0|H, \beta \sim \mathcal{DP}(\beta, H) \tag{1.7}$$

with base distribution H and concentration parameter β. G_0 is the "average" distribution over possibly infinite components $\{\boldsymbol{\theta}_k\}_{k=1}^{\infty}$ across all groups, and these shared components have different probabilities of usage for different groups based on the data clustered.

With different representations of the DP, the HDP can also be implemented with two main inference algorithms, the Chinese restaurant franchise (Teh et al., 2004b) and the stick-breaking construction (Wang et al., 2011). The HDP has been applied

in many fields, such as document analysis (Wang et al., 2011), audio analysis (Ren et al., 2008), target tracking (Fox et al., 2007) and gene expression clustering (Wang and Wang, 2013), etc. In Chapter 4 a novel integration of the HDP model and dictionary learning model introduced later will be covered with the application of image organization.

1.1.3 Beta Process

The beta process is a stochastic process belonging to the *completely random measure* class (Kingman, 1967). Based on (Thibaux and Jordan, 2007a), consider B_0 be a finite and continuous base measure over a probability space $\Omega = \mathbb{R}_+$ with total mass $B_0(\Theta) = \alpha$, we define the following Lévy measure on the product space $[0, 1] \times \Omega$:

$$\nu(d\omega, d\theta) = c\omega^{-1}(1-\omega)^{c-1}d\omega B_0(d\theta) \tag{1.8}$$

where $c > 0$ is the concentration parameter. We denote such a beta process by $BP(c, B_0)$, and a draw $H \sim BP(c, B_0)$ is then denoted as

$$H = \sum_{k=1}^{\infty} \omega_k \delta_{\theta_k} \tag{1.9}$$

where $(\omega_k, \theta_k)_{k=1}^{\infty}$ are the countably infinite atoms corresponding to the locations $\omega_k \in \Omega$ and weight $\theta_k \in [0, 1]$ generated by a Poisson Process with rate measure ν. Let $q_k \in [0, 1]$ denote the weight of the kth atom, then its weight of this atom drawn from $B \sim \mathrm{BP}(c, B_0)$ follows

$$\omega_k \sim \mathrm{Beta}(cq_k, c(1-q_k)) \tag{1.10}$$

Following (Paisley and Carin, 2009a), we further extend the beta process to take two parameters, $a > 0$, $b > 0$ and the base measure H_0, denoted as $\mathrm{BP}(a, b, H_0)$,

which this thesis is based upon. A draw $H \sim \mathrm{BP}(a, b, H_0)$ is represented as

$$H(\omega) = \sum_{k=1}^{K} \pi_k \delta_{\omega_k}(\omega) \tag{1.11}$$

$$\pi_k \sim \mathrm{Beta}(a/K, b(K-1)/K)$$

$$\omega_k \overset{\text{iid}}{\sim} B_0$$

As $K \to \infty$, $H(\omega)$ leads to an infinite dimensional vector of probabilities, each corresponding to an atom ω_k i.i.d drawn from B_0.

Beta process is conjugate to the Bernoulli process BeP(H) with the measure $X_i(\omega) = \sum_k z_{ik}\delta_{\omega_k}$, where $\boldsymbol{z}_i$ is the binary column vector with the kth value drawn from

$$z_{ik} \sim \mathrm{Bernoulli}(\pi_k) \tag{1.12}$$

We draw N such binary vectors, constituting a $N \times K$ binary matrix $\mathbf{Z} = [\boldsymbol{z}_1, ..., \boldsymbol{z}_N]$. The sparseness of this binary matrix $\mathbf{Z}$ will be controlled by a and b. When π_k is marginalized out, the expected number of non-zero elements of $\boldsymbol{z}_i$ equals $aK/(a + b(K-1))$. As $K \to \infty$, the number of non-zero elements is Poisson(a/b) distributed with its expectation of a/b.

1.2 Nonparametric Bayesian Dictionary Learning

Recent research on dictionary learning and sparse coding has demonstrated superior performance in a number of challenging image processing applications, including image denoising, inpainting and sparse image modeling Aharon et al. (2006); Mairal et al. (2008a); Zhou et al. (2009). It has been known that learning a dictionary adapted to a specific set of signals can yield significantly better performance than using an off-the-shelf dictionary or basis, such as the discrete cosine transform (DCT) or wavelet basis Elad and Aharon (2006). Typically, a good dictionary is "similar" to

the data in the sense that each signal can be well approximated by a linear combination of a few dictionary atoms. Finding such a representation, given a dictionary, is the goal of *sparse coding* Chen et al. (1999); Olshausen and Field (1997). Dictionary learning and sparse coding are often jointly employed for many applications, such as image denoising Elad and Aharon (2006) and interpolation/inpainting Mairal et al. (2008c). Numerous methods are available, for instance Elad and Aharon (2006); Lee et al. (2007); Mairal et al. (2008c); Zhou et al. (2012, 2009).

Let $\boldsymbol{x}_i \in \mathbb{R}^P$ represent the ith data sample and $\{\boldsymbol{x}_i\}_{i=1,N}$ represents the complete data set under analysis. For the application considered here, each $\boldsymbol{x}_i$ corresponds to a set of contiguous pixels (from a small image "patch" extracted from an overall image). The set $\{\boldsymbol{x}_i\}_{i=1,N}$ represents data extracted from N image patches, across all images of interest. Each $\boldsymbol{x}_i$ is assumed to be represented as a linear combination of a sparse set of atoms from a dictionary $\mathbf{D} \in \mathbb{R}^{P\times K}$, where the columns of $\mathbf{D}$ represent dictionary atoms. A prior is placed on $\mathbf{D}$, and a posterior density function on $\mathbf{D}$ is learned based on $\{\boldsymbol{x}_i\}_{i=1,N}$. Further, the size of the dictionary (total number of *active* atoms across all $\boldsymbol{x}_i$) is unknown, and to be inferred; *i.e.*, it is anticipated that only a subset of the K candidate dictionary elements are used. Specifically, for each i, $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{\alpha}_i + \boldsymbol{\epsilon}_i$, where $\boldsymbol{\alpha}_i \in \mathbb{R}^K$ is sparse and $\|\boldsymbol{\epsilon}_i\|_2/\|\boldsymbol{x}_i\|_2 \ll 1$. Additionally, a prior is placed on $\{\boldsymbol{\epsilon}_i\}_{i=1,N}$, and the statistics of the residual are also to be inferred.

1.2.1 Beta Process Factor Analysis

In recent research Zhou et al. (2009), it has been demonstrated that the beta process (BP) and Bernoulli process (BeP) may be coupled to constitute a prior on $\{\boldsymbol{\alpha}_i\}_{i=1,N}$ and $\mathbf{D}$, to impose the desired sparseness and to infer the dictionary composition and size; this construction also imposes that many of the $\boldsymbol{x}_i$ will use a similar subset of columns of $\mathbf{D}$.

We represent each image as a set of local patches as $\{\boldsymbol{x}_i\}_{i=1,N}$, where N represents

the total number of patches in this image, and $\boldsymbol{x}_i$ is the data from the ith patch. We use Bayesian dictionary learning on the data $\{\boldsymbol{x}_i\}_{i=1,N}$ to infer a dictionary $\mathbf{D}$ under which each $\boldsymbol{x}_i$ is sparsely represented. Specifically, each $\boldsymbol{x}_i$ is represented as

$$\boldsymbol{x}_i = \mathbf{D}(\boldsymbol{z}_i \odot \boldsymbol{s}_i) + \epsilon_i \tag{1.13}$$

where $\odot$ represents the pointwise/Hadamard vector product, K is the truncation level on the possible number of dictionary atoms, $\boldsymbol{z}_i = [z_{i1}, \cdots, z_{iK}]^T$, $\boldsymbol{s}_i = [s_{i1}, \cdots, s_{iK}]^T$, $z_{ik} \in \{0, 1\}$ indicates whether the kth atom is *active* within patch i in image, $s_{ik} \in \mathbb{R}$, and $\boldsymbol{\epsilon}_i$ is the residual error. Note that under an appropriate dictionary $\mathbf{D}$, $\boldsymbol{z}_i$ represents the specific sparseness pattern of dictionary usage for $\boldsymbol{x}_i$. This part of the model is as in previous Bayeasian dictionary learning Zhou et al. (2009), and the unique component of the model is to link the sparse binary vector $\boldsymbol{z}_i$ to a topic model.

Let the binary vector $\boldsymbol{z}_i \in \{0, 1\}^K$ denote which of the K columns of $\mathbf{D}$ are used for representation of $\boldsymbol{x}_i$ (active set); if a particular component of $\boldsymbol{z}_i$ is equal to one, then the corresponding column of $\mathbf{D}$ is used in the representation of $\boldsymbol{x}_i$. Hence, for the data $\{\boldsymbol{x}_i\}_{i=1,N}$ there is an associated set of latent binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$, and the beta-Bernoulli process provides a convenient prior for these vectors (Paisley and Carin, 2009a; Thibaux and Jordan, 2007b; Zhou et al., 2009). Specifically, consider the model

$$\boldsymbol{z}_i \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_k) \ , \quad \boldsymbol{\pi} \sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \tag{1.14}$$

where π_k is the kth component of $\boldsymbol{\pi}$, and a and b are model parameters; the impact of these parameters on the model are discussed below. Note that the use of product notation in (5.2) is meant to denote that each component of $\boldsymbol{z}_i$ and $\boldsymbol{\pi}$ are drawn independently from distributions of the same form.

Considering the limit $K \rightarrow \infty$, and after integrating out $\boldsymbol{\pi}$, the draws of $\{\boldsymbol{z}_i\}_{i=1,N}$ may be constituted as follows. For each $\boldsymbol{z}_i$, draw $c_i \sim \text{Poisson}(\frac{a}{b+i-1})$ and define $C_i = \sum_{j=1}^{i} c_j$, with $C_0 = 0$. Let z_{ik} represent the kth component of $\boldsymbol{z}_i$, and $z_{ik} = 0$ for $k > C_i$. For $k = 1, \ldots, C_{i-1}$, $z_{ik} \sim \text{Bernoulli}(\frac{n_{ik}}{b+i-1})$, where $n_{ik} = \sum_{j=1}^{i-1} z_{jk}$ (n_{ik} represents the total number of times the kth component of $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is one). For $k = C_{i-1} + 1, \ldots, C_i$, we set $z_{ik} = 1$. Note that as $a/(b + i - 1)$ becomes small, with increasing i, it is probable that c_i will be small. Hence, with increasing i, the number of new non-zero components of $\boldsymbol{z}_i$ diminishes. Further, as a consequence of $\text{Bernoulli}(\frac{n_{ik}}{b+i-1})$, when a particular component of the vectors $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is frequently one, it is more probable that it will be one for subsequent $\boldsymbol{z}_j$, $j \geq i$. When $b = 1$ this construction for $\{\boldsymbol{z}_i\}_{i=1,N}$ corresponds to the Indian buffet process (Griffiths and Ghahramani, 2005b).

Since $\boldsymbol{z}_i$ defines which columns of $\mathbf{D}$ are used to represent $\boldsymbol{x}_i$, (5.2) imposes that it is probable that some columns of $\mathbf{D}$ are used repeatedly among the set $\{\boldsymbol{x}_i\}_{i=1,N}$, while other columns of $\mathbf{D}$ may be more specialized to particular $\boldsymbol{x}_i$. As demonstrated below, this has been found to be a good model when $\{\boldsymbol{x}_i\}_{i=1,N}$ are patches of pixels extracted from natural images.

The hierarchical form of the model may now be expressed as

$$\begin{aligned} \boldsymbol{x}_i &= \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i \\ \boldsymbol{w}_i &= \boldsymbol{z}_i \odot \boldsymbol{s}_i \\ \boldsymbol{d}_k &\sim \mathcal{N}(0, P^{-1}\mathbf{I}_P) \\ \boldsymbol{s}_i &\sim \mathcal{N}(0, \gamma_s^{-1}\mathbf{I}_K) \\ \boldsymbol{\epsilon}_i &\sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P) \end{aligned} \tag{1.15}$$

where $\boldsymbol{d}_k$ represents the kth component (atom) of $\mathbf{D}$, $\odot$ represents the elementwise or Hadamard vector product, $\mathbf{I}_P$ ($\mathbf{I}_K$) represents a $P \times P$ ($K \times K$) identity matrix,

and $\{\boldsymbol{z}_i\}_{i=1,N}$ are drawn as in (5.2). Conjugate hyperpriors $\gamma_s \sim \text{Gamma}(c, d)$ and $\gamma_\epsilon \sim \text{Gamma}(e, f)$ are also imposed. The construction in (3.1), and with the prior in (5.2) for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as the beta process factor analysis (BPFA) model. This model was first developed in (Knowles and Ghahramani, 2007), with a focus on general factor analysis; here we apply and extend this construction for image-processing applications.

Note that we impose independent Gaussian *priors* for $\boldsymbol{d}_k$, $\boldsymbol{s}_i$ and $\boldsymbol{\epsilon}_i$ for modeling convenience (conjugacy of consecutive terms in the hierarchical model). However, the inferred *posterior* for these terms is generally *not* independent or Gaussian. The independent priors essentially impose prior information about the *marginals* of the posterior of each component, while the inferred posterior accounts for statistical dependence as reflected in the data.

1.3 Topic Modeling on Imagery and Text Analysis

Living in such an information explosion era with rapid growth of data everywhere, we are exposed to massive news, articles, blogs and emails posted through newspaper, internet, and social medias and mailboxes every day. Machine learning researchers then have developed probabilistic topic modeling, aiming to uncover and summarize the themes of large archives of documents in unsupervised way.

1.3.1 Topic Models

One typical example of such algorithms is Latent Dirichlet Allocation (LDA). The basic idea of LDA is that documents are represented as mixtures over latent *topics*, each of which characterizes an underlying semantic theme. Each topic is represented as one specific distribution over terms in a vocabulary. Given a corpus of documents to analyze, LDA can eventually infer how topics are represented and how each document is modelled as deriving from a subset of topics. Such descriptive statistics for

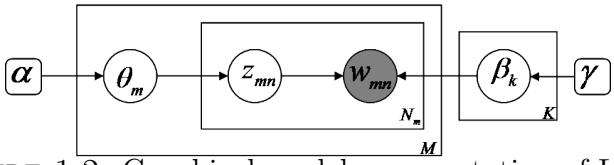


FIGURE 1.2: Graphical model representation of LDA.

this corpus could help us do browsing, prediction, and searching.

To be specific, we have a corpus of M documents $\mathbf{D} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_M\}$, and $\boldsymbol{w}_m = (w_{m1}, w_{m2}, \ldots, w_{mN_m})^T$ represents the mth document, where w_{mn} denotes the nth word in this mth document. N_m is the total number of words in document m, and K is pre-defined truncation number of topics. The LDA is represented as a probabilistic graphical model in 1.2. For document m,the general process of LDA can be summarized as

1. For topic $k = 1, \ldots, K$
   - Draw $\boldsymbol{\beta}_k \sim \text{Dir}(\gamma)$
2. For each document $m = 1, \ldots, M$
   - Draw $N_m \sim \text{Poisson}(\zeta)$
   - Draw topic proportion $\theta_m \sim \text{Dir}(\alpha)$
   - For each word $w_{mn}, m = 1, \ldots, N_m$
     (a) Draw a topic indicator $z_{mn} \sim \text{Discrete}(\theta_m)$
     (b) Draw a word w_{mn} from $p(w_{mn}|z_{mn}, \boldsymbol{\beta})$, a multinomial probability $\boldsymbol{\beta}_{z_{mn}}$

Such topic models are unsupervised that only words in the documents are considered. Besides purely collection of documents, some other types of data could also be modeled based on probabilistic topic models. Researchers developed supervised topic models (Blei and MaAuliffe, 2007) to infer latent topics predictive of the response.

Such document-response corpora could be documents with their text categories, essays with their grades, movie reviews with their numerical ratings and web pages with counts of online community members who like them, etc.

Furthermore, (Iwata et al., 2009) proposed a probabilistic topic modeling for analyzing and detecting content-related annotations from noisy annotated data. They introduce an indicator variable judging whether each annotation is related to the content or not. If so, this annotation term is generated from the topics that generated the content, otherwise from a content-unrelated general pool. This proposed model has successfully applied in several real social annotation data taken from social bookmarking services and photo sharing services.

1.3.2 Hierarchical Topic Models

Previous topic models could effectively find useful set of topics from a given corpus of documents, researchers further hope to organize the topics via a hierarchy. (Blei et al., 2010, 2003a) proposes the nested Chinese restaurant process (nCRP) as a prior distribution to infer the hierarchical tree structure of topics. In this tree structure, each node denotes one topic. Each document belongs to one path, in which way each document is treated as a mixture model of the several topics only along its own path, rather than all latent topics in (Blei et al., 2003b). Since the nodes near the root of the tree will be shared more than the ones close to the leaves, topics are naturally formed in a general-to-specific hierarchy that more abstract topics are in the top and more concrete topics are in the bottom. Its generative process is summarized in the following:

1. For node $k \in \mathcal{T}$ in the infinite tree
    - Draw $\boldsymbol{\beta}_k \sim \text{Dir}(\gamma)$
2. For each document $m = 1, \ldots, M$

- Draw the path $\boldsymbol{c}_m \sim \text{nCRP}(\zeta)$
- Draw a distribution over levels in the tree $\theta_m \sim \text{DP}(\alpha)$
- For each word $w_{mn}, m = 1, \ldots, N_m$
  (a) Draw a level indicator $z_{mn} \sim \text{Discrete}(\theta_m)$
  (b) Draw a word w_{mn} from $\text{Discrete}(w_{mn}|z_{mn}, \boldsymbol{\beta})$, a multinomial probability $\boldsymbol{\beta}_{z_{mn}}$ parameterized by the topic in the position z_{mn} along the path $\boldsymbol{c}_m$

This BNP approach is able to infer the width of the infinite tree by pre-assigning fairly large truncation numbers of associated children nodes at each level, without choosing the topology of the hierarchy in advance. Effective results on several collections of documents have shown the advantage of nCRP (Blei et al., 2010, 2003a). Such nCRP, originally implemented via MCMC sampling (Blei et al., 2010, 2003a), is later employed based on variational methods in (Wang and Blei, 2009). To be specific, implementing DP via the stick-breaking construction 1.3, (Wang and Blei, 2009) develops an alternative tree-based stick-breaking construction of nCRP mixture model, which facilitates the derivation of variational method. Besides text, (Wang and Blei, 2009) also demonstrates its ability to model continuous data, i.e., simple hand written digits.

1.3.3 Review of Topic Models for Imagery

Statistical topic models, originally developed for text analysis, have also been applied successfully for image-analysis tasks. In this setting researchers typically represent an image as a bag of visual words Fei-Fei and Perona (2005); Li and Fei-Fei (2007). Using such methods, there has been interest in developing models for automatic clustering, classification and annotation of images, based on image features as well as available meta-data such as image annotations (Barnard et al., 2003; Blei and

Jordan, 2003; Blei and MaAuliffe, 2007; Du et al., 2009; Li et al., 2009; Wang et al., 2009).

In such research one typically treats image feature extraction as a pre-processing step, decoupled from the subsequent statistical analysis. Local image descriptors, *e.g.*, scale-invariant feature transform (SIFT) Lowe (1999), are commonly used to extract features from local patches Fei-Fei and Perona (2005); Li and Fei-Fei (2007); Wang et al. (2009), segments Li et al. (2009), or super-pixels Du et al. (2009). Researchers have also considered hue feature vectors and the output of maximum response (MR) filter banks Du et al. (2009); Ren et al. (2011), shape and location feature extraction Li et al. (2009), among others. Maximum or average pooling operations have been employed to characterize relationships of neighborhood features Lazebnik et al. (2006); Yang et al. (2009). In such research the extracted local features are typically used to design a discrete codebook (*i.e.*, vocabulary), with vector quantization (VQ). When analyzing images, each local descriptor is subsequently assigned to one of the codewords, with these codes playing the role of discrete "words" in traditional documents Fei-Fei and Perona (2005). Following the traditional topic models, the goal is to infer the latent topics over these codes.

Li et al. (2010) has successfully applied nCRP framework on one large-scale challenging real image dataset, aiming to infer a meaningful image hierarchy. Similarly, compared with nCRP (Blei et al., 2010, 2003a), images in Li et al. (2010) are treated as documents, while local patches extracted in each image are then viewed *visual* words. The goal is to learn a tree hierarchy representing these images, with the aid of their associated tags. In this tree, each node represents one *visual* topic, a distribution over visual features extracted from patches. Each image then takes one brunch/path of this potentially infinite tree, and each patch in this image resides one node along this brunch/path. Furthermore, each node also denotes one *verbal* topic, another distribution over vocabulary of tags. Such framework is able to automati-

cally infer the "semantivisual" image hierarchy by jointly modeling both image and tag information into a general-to-specific image relationship. Both qualitative and quantitative results have shown the effectiveness of the proposed model.

Although the above research has realized significant success, there is no principled way to define the codebook size, which may cause a loss of information in this quantization step; this parameter must be tuned and is in general a function of the dataset considered (the proper codebook may change with different types of images, and as new imagery are observed). Further, since feature extraction is performed separately from the subsequent statistical analysis, it is unclear which features should be used and why one class of features should be preferred. Recent advances in image classification show that substantially improved performance may be achieved by extracting features from local descriptors with dictionary learning and sparse coding, this replacing VQ (Wang et al., 2010; Yang et al., 2009). However, it is not clear how to integrate these tools with topic modeling, to constitute an overall statistical model.

Motivated by these drawbacks, Chapter 4 proposes a novel nonparametric Bayesian model to integrate feature extraction with model learning together into a unified framework, for joint analysis of multiple images and (when present) associated annotations. Under this framework features will be learned and adjusted from the topic modeling part and vice versa. Instead of using a single code as VQ, multiple dictionary atoms will be used to sparsely represent an image patch, which provides the robustness to the dictionary size. An efficient Gibbs slice sampling algorithm is applied for inference, therefore the dictionary atoms/codebook will keep updating to better fit the dataset. This proposed model clusters images into groups at the same level, as a "flat" clustering. Chapter 5 extends this "flat" clustering assumption to a hierarchical dictionary learning model for joint analysis of imagery and associated text. The proposed model assigns probability distributions to ensembles of infinitely

deep, infinitely branching hierarchical trees. Similar to (Blei et al., 2010), each node in the tree denotes one topic. Images are modeled as paths down a random tree, leading to clustering of images according to sharing of topics at multiple levels of tree. Unlike the independent topics in the "flat" clustering model described in Chapter 4, one general-to-specific relationship are automatically imposed among topics residing from top to bottom of the tree. By imposing this assumption allows for more accurate topics to be inferred.

1.4 Thesis Organization

The main contribution of this thesis is to develop novel nonparametric Bayesian models based on dictionary learning for joint imagery and text analysis. The remaining chapters are organized as follows, including methodologies and results.:

Chapter 2 extends the beta process factor analysis (BPFA) model described in Section 1.2 by employing the probit stick-breaking process (PSBP) (Rodriguez and Dunson, 2011) to impose spatial inter-relationships within imagery. This idea is motivated by the fact that the probability vectors of dictionary atoms used for image patches are assumed to be similar within a particular segment class. Instead of one such globally shared probability vector in BPFA model, multiple ones are proposed, each representing one segment type. Thus every image patch will select one probability vector that is mostly "close" to itself, yielding image segmentation. Again the nonparametric prior imposed upon the probabilities of clusters allows for inferring the actual number of segment types used. Using the PSBP encourages proximate and similar patches to be clustered within the same segment type. Joint with dictionary learning, this model could perform image segmentation and interpretation simultaneously for a corrupted image. Inpainting examples of both gray-scale and RGB images are presented to prove the assumption that the extra spatial information imposed could further reduce the reconstruction error, with smooth image

segmentation results.

Chapter 3 provides two techniques of Bayesian Dictionary Learning for large datasets based on the model described in Section 1.2. One is to develop an online variational Bayesian (VB) algorithm, based on online stochastic optimization with a natural gradient step. The other one is to develop parallel framework using Map-Reduce paradigm, implemented on the hadoop server. Instead of processing the entire dataset in a batch version, both methods analyze only a small subset of the huge dataset, so as to handle massive datasets, including those arriving in a stream. State-of-the-art performance is demonstrated by experiments with large natural images containing tens of millions of pixels.

Chapter 4 develops a new nonparametric Bayesian model to integrate Dictionary Learning described in Section 1.2 and topic models into a unified framework. The model assumes that each type/category of images owns one unique distribution over objects/topics, which are the key links to the dictionary learning part. Each object/topic corresponds to one unique sparse probability vector of dictionary atoms. This sparseness is imposed by a truncated beta-Bernoulli process prior. The model is employed to analyze partially annotated images, with the dictionary learning performed directly on image patched. Under the nonparametric framework, the number of image categories, the number of objects, the number of dictionary atoms to be used and how dictionary atoms are constructed could automatically be data-driven inferred. Efficient inference is performed with a Gibbs-slice sampler, and encouraging results are reported on four widely used datasets.

Chapter 5 extends the flat clustering model discussed in Chapter 4 to a hierarchical tree structure. A tree-based dictionary learning model is developed for joint analysis of imagery and associated text (when available). The dictionary learning may be applied directly to the imagery from patches, or to general feature vectors extracted from patches or super-pixels (using any existing method for image feature

extraction). Each image is associated with a path through the tree (from root to a leaf), and each of the multiple patches in a given image is associated with one node in that path. Nodes near the tree root are shared between multiple paths, representing image characteristics that are common among different types of images. Moving toward the leaves, nodes become specialized, representing details in image classes. If available, words (text) are also jointly modeled, with a path-dependent probability over words. The tree structure is inferred via a nested Dirichlet process, and a retrospective stick-breaking sampler is used to infer the tree depth and width. Example results are presented on several datasets with the learned hierarchical structures. From the comparisons to the model described in Chapter 4 and other models, the proposed hierarchical model fits the data better and performs better on the average classification accuracy.

Chapter 6 summarizes this thesis and its contributions. Several possible directions are also discussed for future work.

2

Nonparametric Image Interpolation and Dictionary Learning using Probit Stick-Breaking Process Priors

In image analysis there is often additional information that may be exploited when learning dictionaries, with this well suited for Bayesian priors. For example, most natural images may be segmented, and it is probable that dictionary usage will be similar for regions within a particular segment class. To address this idea, we extend the model by employing a Probit Stick-Breaking Process (PSBP), with this a generalization of the Dirichlet process (DP) stick-breaking representation (Sethuraman, 1994a). Related clustering techniques have proven successful in image processing (Mairal et al., 2009b). The model clusters the image patches, with each cluster corresponding to a segment type; the PSBP encourages proximate and similar patches to be included within the same segment type, thereby performing image segmentation and dictionary learning simultaneously.

2.1 Introduction

In the model discussed in Section 1.2 each patch $\boldsymbol{x}_i$ had a unique usage of dictionary atoms, defined by the binary vector $\boldsymbol{z}_i$, which selects columns of $\mathbf{D}$. One may wish to place further constraints on the model, thereby imposing a greater degree of statistical structure. For example, one may employ that the $\boldsymbol{x}_i$ cluster, and that within each cluster each of the associated $\boldsymbol{x}_i$ employ the same columns of $\mathbf{D}$. This is motivated by the idea that a natural image may be clustered into different types of textures or general image structure. However, rather than imposing that all $\boldsymbol{x}_i$ within a given cluster use *exactly* the same columns of $\mathbf{D}$, one may want to impose that all $\boldsymbol{x}_i$ within such a cluster share the same probability of dictionary usage, *i.e.*, that all $\boldsymbol{x}_i$ within cluster c share the same probability of using columns of $\mathbf{D}$, defined by $\boldsymbol{\pi}_c$, rather than sharing a single $\boldsymbol{\pi}$ for all $\boldsymbol{x}_i$ (as in the original model above). Again, such clustering is motivated by the idea that natural images tend to segment into different textural or color forms. Below, we perform clustering in terms of the vectors $\boldsymbol{\pi}_c$, rather than explicit clustering of dictionary usage, which would entail cluster-dependent $\boldsymbol{z}_c$; the "softer" nature of the former clustering structure is employed to retain model flexibility, while still encouraging sharing of parameters within clusters.

A question when performing such clustering concerns the *number* of clusters needed, this motivating the use of nonparametric methods, like those considered in the next subsections. Additionally, since the aforementioned clustering is motivated by the segmentations characteristic of natural images, it is desirable to explicitly utilize the spatial location of each image patch, encouraging that the patches $\boldsymbol{x}_i$ in a particular segment/cluster are spatially contiguous. This latter goal motivates use of a probit stick-breaking process, as also detailed below.

2.2 Probit stick-breaking processes

The Dirichlet process (DP) (Ferguson, 1973a) constitutes a popular means of performing nonparametric clustering. For our problem it has proven effective to set

$$G_0 = \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \tag{2.1}$$

analogous to (5.2), and hence $G = \sum_{l=1}^{\infty} \beta_l \delta_{\boldsymbol{\pi}_l^*}$. The $\boldsymbol{\pi}_l^*$, drawn from G_0, correspond to distinct probability vectors for using the K dictionary atoms (columns of $\mathbf{D}$). For sample i we draw $\boldsymbol{\pi}_i \sim G$, and a separate sparse binary vector $\boldsymbol{z}_i$ is drawn for each sample $\boldsymbol{x}_i$, as $\boldsymbol{z}_i \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{ik})$, with π_{ik} the kth component of $\boldsymbol{\pi}_i$. In practice we truncate the infinite sum for G to N_L elements, and impose $V_{N_L} = 1$, such that $\sum_{l=1}^{N_L} \beta_l = 1$. A (conjugate) gamma prior is placed on the DP parameter α.

We may view this DP construction as an "Indian buffet franchise," generalizing the Indian buffet analogy (Griffiths and Ghahramani, 2005b). Specifically, there are N_L Indian buffet restaurants; each restaurant is composed of the same "menu" (columns of $\mathbf{D}$), and is distinguished by different probabilities for selecting menu items. The "customers" $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster based upon which restaurant they go to. The $\{\boldsymbol{\pi}_l^*\}_{l=1,N_L}$ represent the probability of using each column of $\mathbf{D}$ in the respective N_L different buffets. The $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster themselves among the different restaurants in a manner that is consistent with the characteristics of the data, with the model also simultaneously learning the dictionary/menu $\mathbf{D}$. Note that we typically make the truncation N_L large, and the posterior distribution infers the number of clusters actually needed to support the data, as represented by how many β_l are of significant value. The model in (3.1), with the above DP construction for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as DP-BPFA.

The DP yields a clustering of $\{\boldsymbol{x}_i\}_{i=1,N}$, but it does not account for our knowledge

of the location of each patch within the image. It is natural to expect that if $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$ are proximate then they are likely to be constituted in terms of similar columns of $\mathbf{D}$[1]. To impose this information, we employ the probit stick-breaking processes (PSBP)(Rodriguez and Dunson, 2011). A *logistic* stick-breaking process is discussed in detail in (Ren et al., 2011). We employ the closely related probit version here because it may be easily implemented in a Gibbs sampler. We note that while the method in (Ren et al., 2011) is related to that discussed below, in (Ren et al., 2011) the concepts of learned dictionaries and beta-Bernoulli priors were not considered. Another related model, that employs a probit link function, is discussed in (Chung and Dunson, 2009).

We augment the data as $\{\boldsymbol{x}_i, \boldsymbol{l}_i\}_{i=1,N}$, where $\boldsymbol{x}_i$ again represents pixel values from the ith image patch, and $\boldsymbol{l}_i \in \mathbb{R}^2$ represents the two-dimensional location of each patch. We build a spatially dependent mixture model for this data as

$$x_i|\theta_i \sim F(\theta_i), \quad \theta_i \sim G_{l_i}, \quad G_{l_i} = \sum_{j=1}^{\infty} \beta_j(l_i)\delta_{\phi_j}, \quad \phi_j \sim G_0 \tag{2.2}$$

We draw an infinite set of model parameter $\{\phi_j\}_{j=1}^{\infty}$. Each observation x_i is drawn from the distribution $F(\theta_i)$ where θ_i denotes the parameter associated with x_i, with $\theta_i \in \{\phi_j\}_{j=1}^{\infty}$. Note that the distinction between DP 1.4 and PSBP is that in the former the mixture weights $\{\beta_j\}_{j=1,J}$ are independent of spatial position $\boldsymbol{l}$, while the latter explicitly utilizes $\boldsymbol{l}$ within $\{\beta_j(\boldsymbol{l})\}_{j=1,J}$ (and below we impose that $\beta_j(\boldsymbol{l})$ changes smoothly with $\boldsymbol{l}$).

The probability that a particular space-dependent data sample employs a particular model parameter is defined by a spatially-dependent stick-breaking process, represented by a kernel-based probit-regression. We introduce an indicator $\boldsymbol{r}_i =$

[1] Proximity can be modeled as in "spatial proximity," as here developed in detail, or "feature proximity" as in non-local means and related approaches, see (Mairal et al., 2009b) and references therein.

$(r_{i1}, \ldots, r_{i\infty})$ to denote which model parameter is associated with x_i. Here $\{r_{ij}\}_{j=1}^{\infty} \in \{0, 1\}$. x_i is associated with model parameter ϕ_j if $r_{ij} = 1$ and $r_{ij'} = 0$ for $j' < j$. Let $\nu_j(l_i)$ denote the probability that $r_{ij} = 1$, thus $1 - \nu_j(l_i)$ represents the probability that $r_{ij} = 0$. Then the probability that the jth parameter is selected equals

$$\beta_j(l_i) = \Phi(g_j(l_i)) \prod_{j'=1}^{j-1} [1 - \Phi(g_{j'}(l_i))] \tag{2.3}$$

where $\Phi(g) = \int_{-\infty}^{g} \mathcal{N}(x|0, 1)dx$ is the cumulative distribution function for the standard normal distribution, thus $0 < \Phi(g_j(\boldsymbol{l})) < 1$. 2.3 is of the same form of the stick-breaking representation (Sethuraman, 1994a). We set $\Phi_J(\boldsymbol{l}_i) = 1$ and select N_c basis points across spatial domain, and for $j \leq J - 1$, each $g_j(\boldsymbol{l}_i)$ is defined:

$$g_j(\boldsymbol{l}_i) = \omega_{j0} + \sum_{n=1}^{N_c} \omega_{jn} \mathcal{K}(\boldsymbol{l}_i, \hat{\boldsymbol{l}}_n; \psi_j) \tag{2.4}$$

where $\mathcal{K}(\boldsymbol{l}_i, \hat{\boldsymbol{l}}_n; \psi_j)$ is a kernel characterized by parameter ψ_j and $\{\omega_{jn}\}_{n=0,N_c}$ are a *sparse* set of real numbers. We here utilize a radial basis function (RBF) kernel

$$\mathcal{K}(\boldsymbol{l}, \hat{\boldsymbol{l}}_n; \psi_j) = \exp[-\frac{\|\hat{\boldsymbol{l}}_n - \boldsymbol{l}\|_2}{\psi_j}] \tag{2.5}$$

to measure the closeness between location $\boldsymbol{l}_i$ and basis point $\hat{\boldsymbol{l}_n}$. We impose the sparseness on the kernel basis coefficients $\mathbf{W}_j = [\omega_{j0}, \omega_{j1}, \ldots, \omega_{jN_c}]'$ by applying Student-t prior as

$$\omega_{jn} \sim \mathcal{N}(\omega_{jn}|0, \lambda_{jn}^{-1}) \text{Gamma}(\lambda_{jn}|a_0, b_0) \tag{2.6}$$

with (a_0, b_0) is set to favor most λ_{jn} being large (if λ_{jn} is large, a draw $\mathcal{N}(0, \lambda_{jn}^{-1})$ is likely to be near zero, such that most $\{\omega_{jn}\}_{i=0,N_c}$ are near zero). This sparseness-promoting construction is the same as that employed in the relevance vector machine

(RVM) (Tipping, 2001). The indicator variables controlling allocation to components are then drawn from

$$r_{ij} \sim \text{Bernoulli}[\Phi(g_j(\boldsymbol{l}_i))] \tag{2.7}$$

Each $g_j(\boldsymbol{l})$ is encouraged to only be defined by a small set of localized kernel functions, and via the probit link function $\int_{-\infty}^{g_j(\boldsymbol{l})} dx\mathcal{N}(x|0,1)$ the probability $\nu_j(\boldsymbol{l})$ is characterized by localized segments over which the probability $\nu_j(\boldsymbol{l})$ is contiguous and smoothly varying. The $\nu_j(\boldsymbol{l})$ constitute a space-dependent stick-breaking process. Since $\nu_J = 1$, $\sum_{j=1}^{J} \beta_j(\boldsymbol{l}) = 1$ for all $\boldsymbol{l}$.

The PSBP model is relatively simple to implement within a Gibbs sampler. For example, as indicated above, sparseness on ω_{jn} is imposed as in the RVM, and the probit link function is simply implemented within a Gibbs sampler (which is why it was selected, rather than a logistic link function). Finally, we define a finite set of possible kernel parameters $\boldsymbol{\Psi}^* = \{\Psi_j^*\}_{j=1,N_p}$, and a multinomial prior is placed on these parameters, with the multinomial probability vector drawn from a Dirichlet distribution (Ren et al., 2011) (each of the $g_j(\boldsymbol{l})$ draws a kernel parameter from $\{\Psi_j^*\}_{j=1,N_p}$). It is desirable to allow flexibility in the kernel parameter ψ, as this will influence the size of segments that are encouraged (discussed further below). Hence, for each j we draw

$$\psi_j = \Psi_{r_j}^*, \quad r_j \sim \text{Mult}(1/N_p, \ldots, 1/N_p) \tag{2.8}$$

with $\boldsymbol{\Psi}^* = \{\psi_p^*\}_{p=1}^{N_p}$ a library of possible kernel-size parameters; r_j is an index for the one non-zero component of a single draw from $\text{Mult}(1/N_p, ..., 1/N_p)$. We employ a discrete dictionary of kernel sizes ψ because there is not a conjugate prior for imposition of a continuous distribution of kernel parameters. A draw from this hierarchical prior is denoted concisely as $G_{\boldsymbol{l}} \sim \text{PSBP}(G_0, \tau_{01}, \tau_{02}, \boldsymbol{\Psi}^*)$, and model parameters $\{\phi_j\}_{j=1}^{\infty}$ are drawn from the base measure G_0. In practice we usually

truncated the PSBP to J sticks, as in a truncated stick-breaking process. With a truncation level J, if $\beta_j(\boldsymbol{l}_i) = 0$ for all $j = 1, \ldots, J-1$, then $\beta_J(\boldsymbol{l}_i) = 1$ so that $\theta_i = \phi_J$. For convenience, we further introduce a scalar indicator ξ_i to represent which model parameter x_i employs as $\xi_i = \text{argmin}_j(r_{ij} = 1)$, thus $\theta_i = \phi_{\xi_i}$.

2.3 Image Interpolation via PSBP

Recall that we wish to impose that proximate patches within an image are more likely to be composed of the same or similar columns of $\mathbf{D}$. To impose this information, we employ the probit stick-breaking process (PSBP). In the PSBP construction, all aspects of (3.1) are retained, except for the manner in which $\boldsymbol{r}_i$ are constituted. Rather than drawing a single K-dimensional vector of probabilities $\boldsymbol{\pi}$ as in (5.2), we draw a *library* of such vectors:

$$\boldsymbol{\pi}_j \sim \prod_{k=1}^{K} \text{Beta}(a_0/K, b_0(K-1)/K) \quad , \quad j = 1, \ldots, J \tag{2.9}$$

and each $\boldsymbol{\pi}_j$ is associated with a particular segment in the image. One $\boldsymbol{\theta}_i$ is associated with location $\boldsymbol{l}_i$, and drawn

$$\boldsymbol{\theta}_i \sim \sum_{j=1}^{J} \beta_j(\boldsymbol{l}_i)\delta_{\boldsymbol{\pi}_j} \tag{2.10}$$

where $\beta_j(\boldsymbol{l}_i)$ is defined in 2.3 with $\sum_{j=1}^{J} \beta_j(\boldsymbol{l}_i) = 1$ for all $\boldsymbol{l}_i$, and $\delta_{\boldsymbol{\pi}_j}$ represents a point measure concentrated at $\boldsymbol{\pi}_j$. Once $\boldsymbol{\theta}_i$ is associated with a particular $\boldsymbol{x}_i$, the corresponding binary vector $\boldsymbol{z}_i$ is drawn as in the first line of (5.2).

The PSBP constitutes J "Indian buffets", defined by $\{\boldsymbol{\pi}_j\}_{j=1,J}$. The buffets share the same "dishes", defined by the columns of $\mathbf{D}$, and the jth buffet has associated probability of dish usage specified by $\boldsymbol{\pi}_j$. The different $\boldsymbol{\pi}_j$ are here used to represent specific segments in an image, and we note that within a segment the dictionary usage across the patches is similar (defined by binary draws from $\boldsymbol{\pi}_j$). Via PSBP

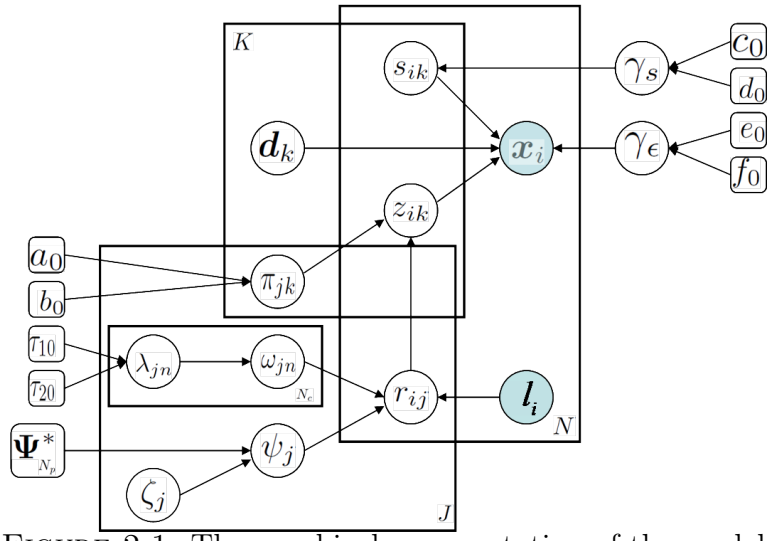


FIGURE 2.1: The graphical representation of the model.

construction, if $\boldsymbol{x}_i$ employs a particular $\boldsymbol{\pi}_j$, its neighboring patch is likely to as well (since $\beta_j(\boldsymbol{l})$ varies smoothly as a function of $\boldsymbol{l}$). However, it is also possible that two widely separated $\boldsymbol{x}_i$ may employ the same $\boldsymbol{\pi}_j$. The integer J sets the truncation level of the spatial stick-breaking process, and the inferred posterior distribution defines how many sticks will have significant weight $\beta_j(\boldsymbol{l})$ for at least some $\boldsymbol{l}$. Hence, if J is set sufficiently large, the algorithm nonparametrically infers the number of segments associated with the image. A graphical representation of the model referred as PSBP-BPFA is summarized in 2.1 with the hierarchical model as follows:

$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{x}_i|\mathbf{D}(\boldsymbol{z}_i \circ \boldsymbol{s}_i), \gamma_\epsilon^{-1}\mathbf{I}_P), \quad \gamma_\epsilon \sim \mathrm{Ga}(e_0, f_0)$$

$$\boldsymbol{d}_k \sim \mathcal{N}(0, 1/P\mathbf{I}_P), \quad s_{ik} \sim \mathcal{N}(s_{ik}|0, \gamma_s^{-1}), \quad \gamma_s \sim \mathrm{Ga}(|c_0, d_0)$$

$$z_{ik} \sim \mathrm{Bernoulli}(\pi_{\xi_i k}), \quad \boldsymbol{\pi}_{jk} \sim \mathrm{Beta}(a0/K, b0(K-1)/K)$$

$$\xi_i \sim \sum_{j=1}^{J} \beta_j(\boldsymbol{l}_i)\delta_j, \quad \beta_j(\boldsymbol{l}_i) = \nu_j(\boldsymbol{l}_i)\prod_{\hat{j}=1}^{j-1}(1-\nu_{\hat{j}}(\boldsymbol{l}_i))$$

$$\nu_j(\boldsymbol{l}_i) = \Phi[g_j(\boldsymbol{l}_i)] \ \text{ for } j = 1, \ldots, J-1, \quad \nu_J(\boldsymbol{l}_i) = 1$$

$$g_j(\boldsymbol{l}_i) = \sum_{n=1}^{N_c} \omega_{jn}\mathcal{K}(\boldsymbol{l}_i, \hat{\boldsymbol{l}}_n; \psi_j) + \omega_{j0}$$

$$\omega_{jn} \sim \mathcal{N}(\omega_{jn}|0, \lambda_{jn}^{-1}), \quad \lambda_{jn} \sim \mathrm{Ga}(\tau_{01}, \tau_{02})$$

$$\psi_j = \Psi^*_{\zeta_j}, \quad \zeta_j \sim \mathrm{Mult}(1/N_p, ..., 1/N_p)$$

where $\mathcal{K}(\boldsymbol{l}_i, \hat{\boldsymbol{l}}_n; \psi_j) = \exp\left(-\frac{||\boldsymbol{l}_i - \hat{\boldsymbol{l}}_n||^2}{\psi_j}\right)$.

To implement PSBP, one must set several parameters. The hyper-parameters associated with the Normal Inverse-Gamma prior on ω_{jn} are set as $\tau_{01} = \tau_{02} = 10^{-6}$, this corresponding to the settings of the related RVM (Tipping, 2001). The number of kernel centers N_c is generally in a natural manner, depending upon the application. Usually, when the image has a lot of textures, more kernel centers may be imposed. The truncation level J on the PSBP may be set to any large value that exceeds the number of segments in the end. We must also define a set of candidate kernel scales, $\{\psi_j^*\}_{j=1}^{N_p}$. These again are set naturally to define the relative range of scales in the data under analysis. Our experience is that any "reasonable" set of kernel widths yields very similar performances.

2.4 Discussion of proposed sparseness-imposing priors

The basic BPFA model is summarized in (3.1), and three related priors have been developed for the sparse binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$: (i) the basic truncated beta-Bernoulli process in (5.2), (ii) a DP-based clustering of the underlying $\{\boldsymbol{\pi}_i\}_{i=1,N}$, and (iii) a PSBP clustering of $\{\boldsymbol{\pi}_i\}_{i=1,N}$ that exploits knowledge of the location of the image patches. For (ii) and (iii), the $\boldsymbol{x}_i$ within a particular cluster have *similar* $\boldsymbol{z}_i$, rather than exactly the same binary vector; we also considered the latter, but this worked less well in practice. As discussed further when presenting results, for denoising and interpolation, all three methods yield comparable performance. However, for CS, (ii) and (iii) yield marked improvements in image-recovery accuracy relative to (i). In anticipation of these results, we provide a further discussion of the three priors on $\{\boldsymbol{z}_i\}_{i=1,N}$ and on the three image-processing problems under consideration.

For the denoising and interpolation problems, we are provided with the data $\{\boldsymbol{x}_i\}_{i=1,N}$, albeit in the presence of noise and potentially with substantial missing pixels. However, for this problem N may be made quite large, since we may consider all possible (overlapping) $B \times B$ patches. A given pixel (apart from near the edges of the image) is present in B^2 different patches. Perhaps because we have such a large quantity of partially overlapping data, for denoising and interpolation we have found that beta-Bernoulli process in (5.2) is sufficient for inferring the underlying relationships between the different data $\{\boldsymbol{x}_i\}_{i=1,N}$, and processing these data collaboratively. However, the beta-Bernoulli construction does not explicitly segment the image, and therefore an advantage of the PSBP-BPFA construction is that it yields comparable denoising and interpolation performance as (5.2), while also simultaneously yielding an effective image segmentation.

For the CS problem, we measure $\boldsymbol{y}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i$, and therefore each of the n measurements associated with each image patch ($\boldsymbol{\Sigma} \in \mathbb{R}^{n \times P}$) loses the original pixels in $\boldsymbol{x}_i$

(the projection matrix $\boldsymbol{\Sigma}$ may also change with each patch, denoted $\boldsymbol{\Sigma}_i$). Therefore, for CS one cannot consider all possible shifts of the patches, as the patches are predefined and fixed in the CS measurement (in the denoising and interpolation problems the patches are defined in the subsequent analysis). Therefore, for CS imposition of the clustering behavior via DP or PSBP provides important information, yielding state-of-the-art CS-recovery results.

2.5 Possible extensions

The "Indian buffet franchise" and probit stick-breaking process constructions considered above and in the results below draw the K-dimensional probability vectors $\boldsymbol{\pi}_l^*$ independently. This implies that within the prior we impose no statistical correlation between the components of vectors $\boldsymbol{\pi}_l^*$ and $\boldsymbol{\pi}_{l'}^*$, for $l \neq l'$. It may be desirable to impose such structure, imposing that there is a "global" probability of using particular dictionary atoms, and the different mixture components within the DP/PSBP constructions correspond to specific draws from global statistics of dictionary usage. This will encourage the idea that there may be some "popular" dictionary atoms that are shared across different mixture components (*i.e.*, popular across different buffets in the franchise). One can impose this additional structure via a *hierarchical* BP construction (HBP) (Thibaux and Jordan, 2007b), related to the *hierarchical* DP (HDP) (Teh et al., 2004b). Briefly, in an HBP construction one may draw the $\boldsymbol{\pi}_l^*$ via the hierarchical construction

$$\boldsymbol{\pi}_l^* \sim \prod_{k=1}^{K} \text{Beta}(c\eta_k, c(1-\eta_k)) \ , \quad \boldsymbol{\eta} \sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \tag{2.11}$$

where η_k is the kth component of $\boldsymbol{\eta}$. The vector $\boldsymbol{\eta}$ constitutes "global" probabilities of using each of the K dictionary atoms (across the "franchise"), and $\boldsymbol{\pi}_l^*$ defines the probability of dictionary usage for the lth buffet. This construction imposes

statistical dependencies among the vectors $\{\boldsymbol{\pi}_l^*\}$.

To simplify the presentation, in the below example results we do *not* consider the HBP construction, as good results have already been achieved with the (truncated) DP and PSBP models discussed above. We note that in these analyses the truncated versions of these models may actually help inference of statistical correlations among $\{\boldsymbol{\pi}_l^*\}$ within the *posterior* (since the set $\{\boldsymbol{\pi}_l^*\}$ is finite, and each vector $\boldsymbol{\pi}_l^*$ is of finite length). If we actually considered the infinite limit on K and on the number of mixture components, inference of such statistical relationships within the posterior may be undermined, because specialized dictionary atoms may be constituted across the different franchises, rather than encouraging *sharing* of highly similar dictionary atoms.

While we do not focus on the HBP construction here, a recent paper has employed the HBP construction in related dictionary learning for image-processing applications, yielding very encouraging results (Zhou et al., 2011b). We therefore emphasize that the basic hierarchical Bayesian construction employed here is very flexible, and may be extended in many ways to impose additional structure.

2.6 Connections to Optimization-Based Methods

Before proceeding to the results, it is of interest to relate the form of the proposed hierarchical Bayesian model to more-traditional approaches that seek a point (single)estimate. Assume we measure $\boldsymbol{x}_i' = Q_{\phi_i}(\boldsymbol{x}_i)$, where ϕ_i denotes the set of pixels observed for $\boldsymbol{x}_i$ (different subsets of pixels are observed for different patches, and by processing all pixels "collaboratively," we may infer the underlying dictionary $\mathbf{D}$ and the sparse $\boldsymbol{w}_i$). The logarithm of the posterior of (all) model parameters $\boldsymbol{\Theta}$, given

observed data $\mathcal{D} = \{\boldsymbol{x}_i'\}_{i=1,N}$ and model hyperparameters $\mathcal{H}$, may be expressed as

$$
\begin{aligned}
-\log\ p(\boldsymbol{\Theta}|\mathcal{D},\mathcal{H}) &= \frac{\gamma_\epsilon}{2}\sum_{i=1}^{N}\|Q_{\phi_i}(\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i\odot\boldsymbol{z}_i))\|_2^2 + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 + \frac{\gamma_s}{2}\sum_{i=1}^{N}\|\boldsymbol{s}_i\|_2^2 \quad (2.12)\\
&- \log f(\{\boldsymbol{z}_i\}_{i=1}^N;\mathcal{H}) - \log\mathrm{Gamma}(\gamma_\epsilon|\mathcal{H}) - \log\mathrm{Gamma}(\gamma_s|\mathcal{H}) + Const.
\end{aligned}
$$

where density function $f(\{\boldsymbol{z}_i\}_{i=1}^N;\mathcal{H})$ represents the particular prior placed on $\{\boldsymbol{z}_i\}_{i=1,N}$, and we have considered the beta-Bernoulli prior, as well as the DP and PSBP constructions. Note that the Gaussian assumption on $\boldsymbol{\epsilon}_i$ yields a Frobenius norm between the observed image data and that manifested by the model (constituted via $\sum_{i=1}^N\|Q_{\phi_i}(\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i\odot\boldsymbol{z}_i))\|_2^2$). Therefore, this construction is closely linked to the optimization approach advocated for near-low-rank matrix completion (Candès and Plan, 2010), which also uses a Frobenius norm. However, rather than an employing an ℓ_1 (Laplacian prior) constraint (Mairal et al., 2009a, 2008b) to impose sparseness on $\boldsymbol{w}_i$, we employ the beta-Bernoulli process and $\boldsymbol{w}_i = \boldsymbol{s}_i\odot\boldsymbol{z}_i$. The beta-Bernoulli process imposes that the binary $\boldsymbol{z}_i$ should be sparse, *and* that there should be a relatively consistent (re)use of dictionary atoms across the image, thereby also imposing self-similarity. Note that consistent use of atoms is encouraged because the active sets are defined by the binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$, and these are all drawn from a shared probability vector $\boldsymbol{\pi}$; this is distinct from drawing the active sets i.i.d. from a Laplacian prior. Further, the beta-Bernoulli prior imposes that many components of $\boldsymbol{w}_i$ are exactly zero, while with a Laplacian prior many components are small but not exactly zero (hence the former is analogous to ℓ_0 regularization, with the latter closer to ℓ_1 regularization).

2.7 Model Inference

Markov chain Monte Carlo (MCMC) (Gilks et al., 1998) is widely used for performing inference with hierarchical models like PSBP. For example, many of the previous

spatially dependent mixtures have been analyzed using MCMC (Duan et al., 2007b; Dunson and Park, 2007; Orbanz and Buhmann, 2008). The H-KSBP (An et al., 2008) model is developed based on a hybrid variational inference inference algorithm; however, nearly half of the model parameters still need to be estimated via a sampling technique. Since all the variables belong to exponential-conjugate family, we simply use Gibbs Sampling to infer all the variables. Below, $\mathbf{\Sigma}_i$ represents the projection matrix on the data, for image patch $\boldsymbol{x}_i$. For the CS problem, $\mathbf{\Sigma}_i$ is typically fully populated, while for the interpolation problem each row of $\mathbf{\Sigma}_i$ is all zeros except for a single one, corresponding to the specific pixel that is measured. The update equations are the conditional probability of each parameter, conditioned on all other parameters in the model. The full likelihood of the hierarchical model can be expressed as

$$P(\mathbf{Y}, \mathbf{\Sigma}, \mathbf{D}, \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\pi}, \gamma_s, \gamma_\epsilon) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i | \mathbf{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{||\mathbf{\Sigma}_i||_0}) \times \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{d}_k | 0, P^{-1} \mathbf{I}_P) \tag{2.13}$$

$$\times \prod_{i=1}^{N} \prod_{k=1}^{K} \mathcal{N}(s_{ik} | 0, \gamma_s^{-1}) \text{Bernoulli}(z_{ik} | \pi_{\xi_i k})$$

$$\times \prod_{i=1}^{N} \prod_{j=1}^{J} \text{Bernoulli}(\xi_i | \Phi[g_j(\boldsymbol{l}_i)])$$

$$\times \prod_{j=1}^{J} \prod_{k=1}^{K} \text{Beta}(\pi_{jk} | a0/K, b0(K-1)/K)$$

$$\times \prod_{j=1}^{J} \prod_{n=1}^{N_c} \mathcal{N}(\omega_{jn} | 0, \lambda_{jn}) \text{Ga}(\lambda_{jn} | \alpha_{01}, \alpha_{02})$$

$$\times \text{Ga}(\gamma_s | c_0, d_0) \text{Ga}(\gamma_\epsilon | e_0, f_0)$$

In order to sample the value of the latent process $\{\xi_i\}_{i=1}^N$ and the weights $\{\boldsymbol{\omega}_j\}_{j=1}^J$, we need to further augment a collection of conditionally independent latent variables $\nu_{ij}^* \sim \mathcal{N}(g_j(\boldsymbol{l}_i), 1)$. Define $\xi_i = j$ if and only if $\nu_{ij}^* > 0$ and $\nu_{ij'}^* < 0$ for $j' < j$, then

the probability

$$Pr(\xi_i = j) = Pr(\nu_{ij}^* > 0, \nu_{ij'}^* < 0 \text{ for } j' < j)$$
$$= \nu_j(\boldsymbol{l}_i) \prod_{j'<j} (1 - \nu_{j'}(\boldsymbol{l}_i)) = \beta_j(\boldsymbol{l}_i) \tag{2.14}$$

Conditionally on this latent variables ν_{ij}^*, this data augmentation scheme allow us to easily compute the augmented variables by sampling from its full conditional distribution

$$\nu_{ji} = \begin{cases} 1, \nu_{ij}^* > 0 \\ 0, \nu_{ij}^* < 0 \end{cases}$$

where $\nu_{ij}^* \sim \mathcal{N}(g_j(\boldsymbol{l}_i), 1)$.

The detailed Gibbs sampling inference for the model variables are listed in Appendix A.

2.8 Experiments

2.8.1 Parameter settings

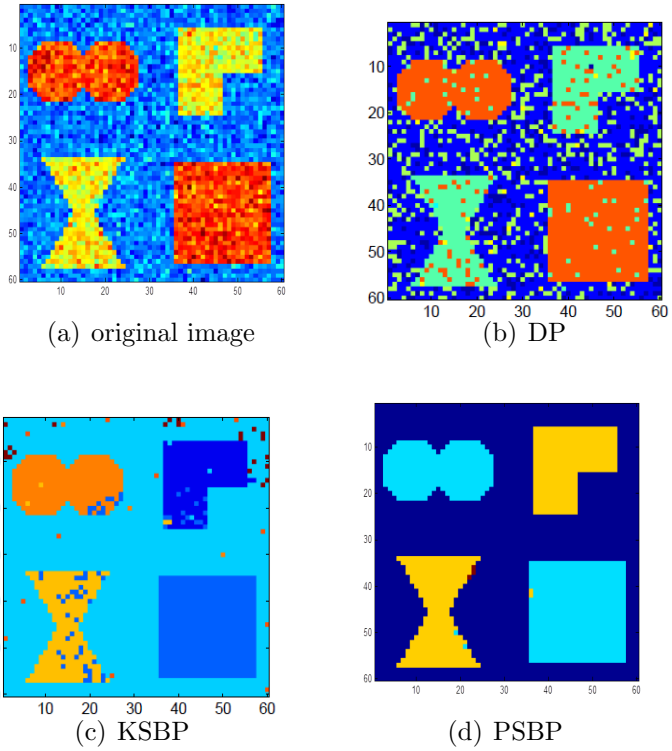
For all BPFA, DP-BPFA and PSBP-BPFA computations, the dictionary truncation level was set at $K = 256$ or $K = 512$ based on the size of the image. Not all K dictionary atoms are used in the model; the truncated beta-Bernoulli process infers the subset of dictionary atoms employed to represent the data $\{\boldsymbol{x}_i\}_{i=1,N}$. The larger the image, the more distinct types of structure are anticipated, and therefore the more dictionary atoms are likely to be employed; however, very similar results are obtained with $K = 512$ in all examples, just with more dictionary atoms not employed for smaller images (therefore, to save computational resources, we set $K = 256$ for the smaller images). The number of DP and PSBP sticks was set at $N_L = 20$. The library of PSBP parameters is defined as in (Ren et al., 2011); the PSBP

kernel locations, $\{\boldsymbol{r}_i\}_{i=1,N}$, were situated on a uniformly sampled grid in each image dimension, situated at every fourth pixel in each direction (the results are insensitive to many related definitions of $\{\boldsymbol{r}_i\}_{i=1,N}$). The hyperparameters within the gamma distributions were set as $c = d = e = f = 10^{-6}$, as is typically done in models of this type (Tipping, 2001) (the same settings were used for the gamma prior for the DP precision parameter α). The beta-distribution parameters are set as $a = K$ and $b = 1$ if random initialization is used or $a = K$ and $b = N/8$ if a singular value decomposition (SVD) based initialization is used. None of these parameters have been optimized or tuned. When performing inference, all parameters are initialized randomly (as a draw from the associated prior) or based on the SVD of the image under test. The Gibbs samplers for the BPFA, DP-BPFA and PSBP-BPFA have been found to mix and converge quickly, producing satisfactory results with as few as 20 iterations. The inferred images represent the average from the collection samples. All software was written in non-optimized Matlab. On a Dell Precision T3500 computer with a 2.4 GHz CPU, for $N = 148,836$ patches of size $8 \times 8 \times 3$ with 20% of the RGB pixels observed at random, the BPFA required about 2 minutes per Gibbs iteration (the DP version was comparable), and PSBP-BPFA required about 3 minutes per iteration. For the 106-band hyperspectral imagery, which employed $N = 428,578$ patches of size $4 \times 4 \times 106$ with 2% of the voxels observed uniformly at random, each Gibbs iteration required about 15 minutes.

2.8.2 Simulation example

In this example the feature vector $\boldsymbol{x}_i$ is the intensity value of each pixel, and the pixel location is the spatial information $\boldsymbol{l}_i$. Each observation is assumed to be drawn from a spatially dependent Gaussian mixture. A comparison is made between the proposed PSBP, the Dirichlet process (DP), and the kernel stick-breaking process (KSBP), and in all cases Gibbs inference is performed; for the KSBP, we use the same model

as considered in (An et al., 2008), and this simple example was also taken from that paper. The data are shown in Figure 6(a), in which four distinct contiguous sub-regions reside in a background, with a color bar encoding the pixel amplitudes. Each pixel is drawn from a Gaussian distribution with a standard deviation of 10; the two pairs of contiguous regions are generated respectively from the Gaussian distributions with mean intensities equal to 40 and 60, and the background has a mean of 5 (An et al., 2008). In the PSBP, DP, and KSBP analysis, we do not set the number of clusters a priori and the models infer the number of clusters automatically from the data. Therefore, we fixed the truncation level to $J = 10$ for all models, and the clustering results are shown in Figure 6, with different colors representing the cluster index (mixture component to which a data sample is assigned).



(a) original image (b) DP

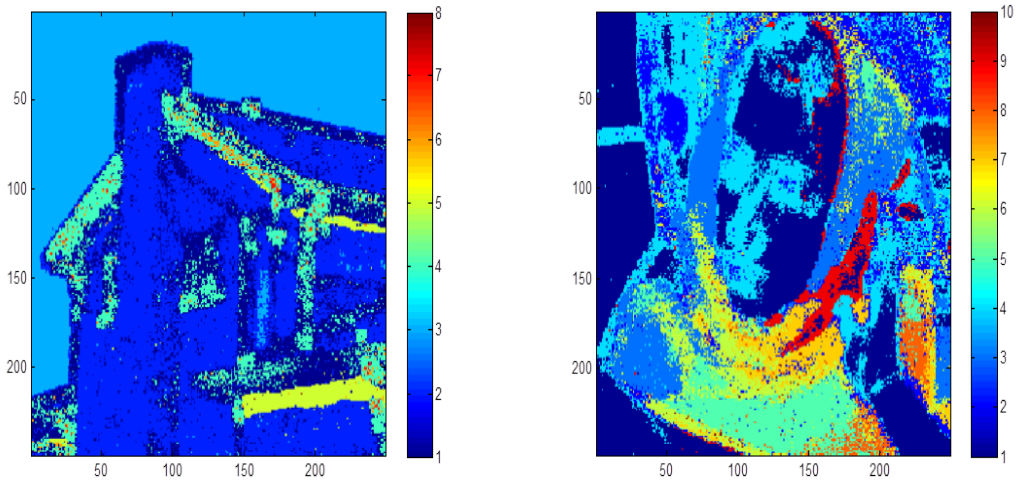
(c) KSBP (d) PSBP

FIGURE 2.2: Segmentation results for the simulation example

Compared with DP and KSBP, the proposed PSBP shows a much cleaner seg-

mentation in Figure 6(d), as a consequence of the imposed favoring of contiguous segments. We also note that the proposed model inferred that there were only three important j (three dominant sticks) within the observed data, consistent with the representation in Figure 6(a)

2.8.3 Gray-scale Image interpolation



(a) House (b) Barbara

FIGURE 2.3: Segmentation results for the gray-scale images, both with 50% data missing.

We then consider standard gray-scale images, with 50% and 80% of the pixels missing uniformly at random. Results are presented for the House and Barbara images , with comparisons between the BPFA model in 3.1 and the BPFA-PSBP model discussed in **??**. For both cases, the truncation level of layers J is set to be 15, and only 8 and 10 layers are actually inferred for the House and Barbara images respectively. We also set a candidate dictionary for kernel widths as $\{600, 700, 800, 900, 1000\}$. The results are shown in 2.3, presenting the mean inferred segmentations. Each color in the inferred segmentation represents one PSBP mixture component, and both

figures show the last Gibbs iteration (to avoid issues with label switching between Gibbs iterations). The comparison of PSNR results are summarized in Table 2.1, from where we could conclude that the proposed PSBP-BPFA performs consistently better than the original BPFA. This results are under our expectation due to the assumption that patches nearby share similar dictionary components rather than independently.

Table 2.1: Comparison of interpolation of gray-scale images, using patch size 8×8 by BPFA and PSBP-BPFA separately.

data ratio	House 20%	House 50%	Barbara 20%	Barbara 50%
BPFA	30.12	38.02	24.80	33.17
PSBP-BPFA	31.25	38.56	25.29	35.94

2.8.4 Color Image interpolation

For the initial interpolation examples, we consider standard RGB images, with 80% of the RGB pixels missing uniformly at random (the data under test are shown in Figure 2.4). Results are first presented for the Castle and Mushroom images, with comparisons between the BPFA model in (5.2) and the PSBP-BPFA model discussed in Section 2. The difference between the two is that the former is a "bag-of-patches" model, while the latter accounts for the spatial locations of the patches. Further, the PSBP-BPFA simultaneously performs image recovery and segmentation. The results are shown in Figure 2.6, presenting the mean reconstructed images and inferred segmentations. Each color in the inferred segmentation represents one PSBP mixture component, and the figure shows the last Gibbs iteration (to avoid issues with label switching between Gibbs iterations). While the BPFA does not directly yield a segmentation, its PSNR results are comparable to those inferred by PSBP-BPFA, as summarized in Table 2.2.

Table 2.2: Comparison of interpolation of the Castle and Mushroom images, based upon observing 20% of the pixels, selected uniformly at random. Results are shown using BPFA and PSBP-BPFA, and the analysis is separately performed using $8\times8\times3$ and $5 \times 5 \times 3$ image patches.

	Castle $8 \times 8 \times 3$	Castle $5 \times 5 \times 3$	Mushroom $8 \times 8 \times 3$	Mushroom $5 \times 5 \times 3$
BPFA	29.32	28.48	31.63	31.17
PSBP-BPFA	29.54	28.46	32.03	31.27

An important additional advantage of Bayesian models like BPFA, DP-BPFA and PSBP-BPFA is that they provide a measure of confidence in the accuracy of the inferred image. In Figure 2.7 we plot the variance of the inferred error $\{\boldsymbol{\epsilon}_i\}_{i=1,N}$, computed via the Gibbs collection samples.

FIGURE 2.4: Images with 80% of the RGB pixels missing at random. Although only 20% of the actual pixels are observed, in these figures the missing pixels are estimated based upon averaging all observed neighboring pixels within a 5×5 spatial extent. Left: castle image (PSNR 22.58 dB), right: mushroom image (24.85 dB).

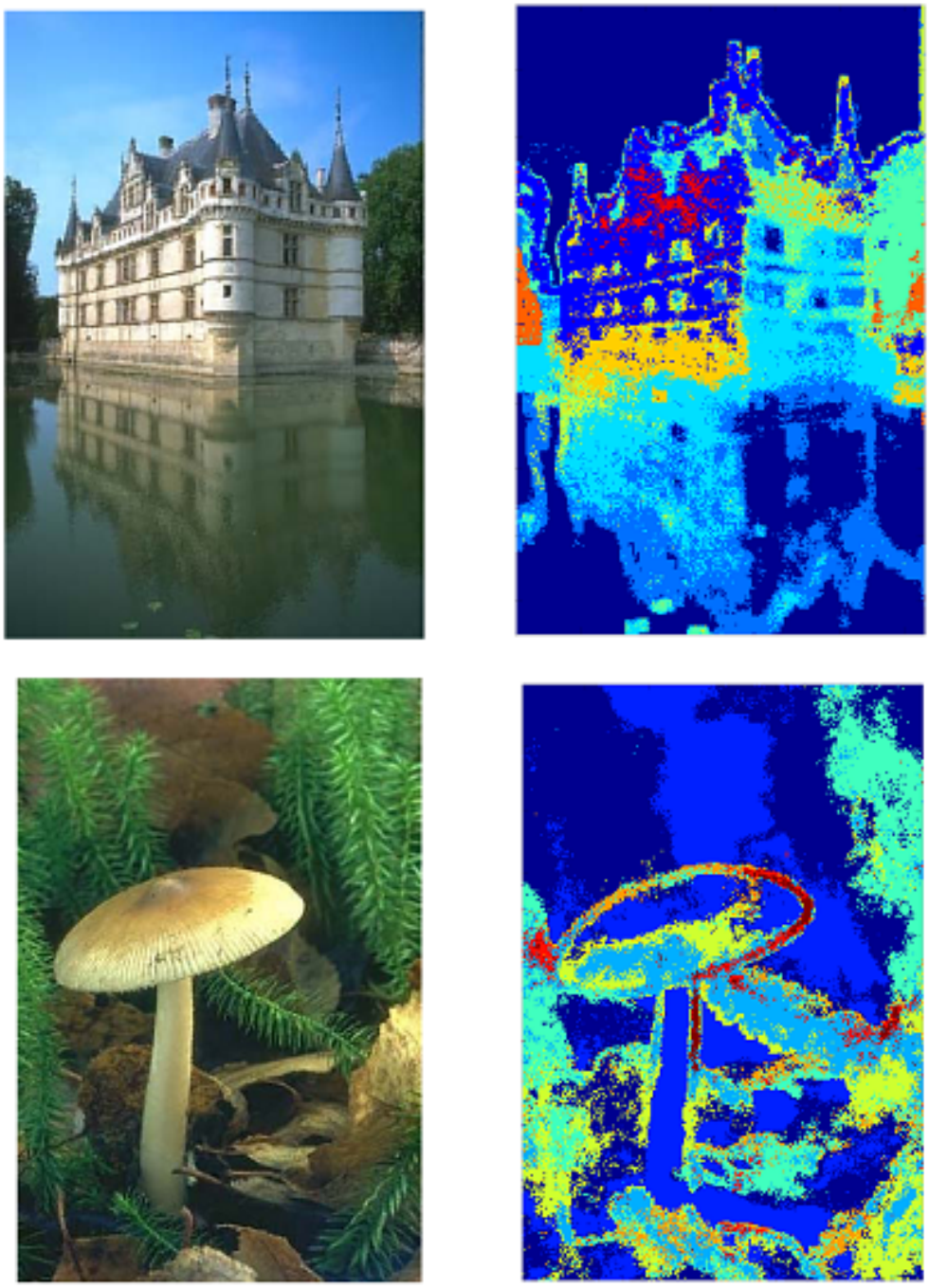


FIGURE 2.5: PSBP-BPFA analysis with 80% of the RGB pixels missing uniformly at random (see Figure 2.4). The analysis is based on $8 \times 8 \times 3$ image patches, considering all possible (overlapping) parches. For a given pixel, the results are the average based upon all patches in which it is contained. For each example, recovered image based on an average of Gibbs collection samples (top), and each color representing one of the PSBP mixture components (bottom).

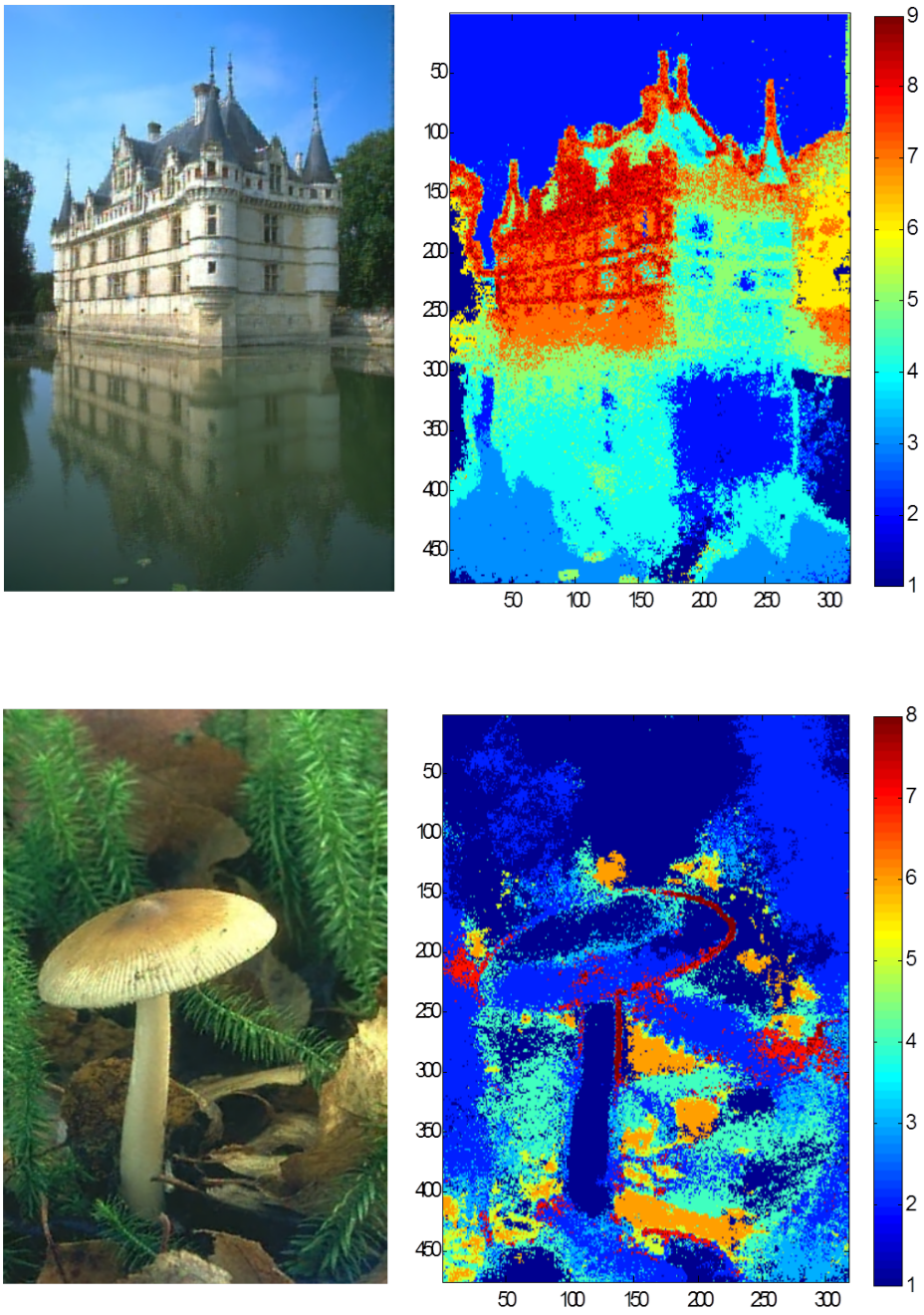


FIGURE 2.6: PSBP-BPFA analysis with 80% of the RGB pixels missing uniformly at random (see Figure 2.4). The analysis is based on $5 \times 5 \times 5$ image patches, considering all possible (overlapping) parches. For a given pixel, the results are the average based upon all patches in which it is contained. For each example, recovered image based on an average of Gibbs collection samples (top), and each color representing one of the PSBP mixture components (bottom).

2.9 Conclusions

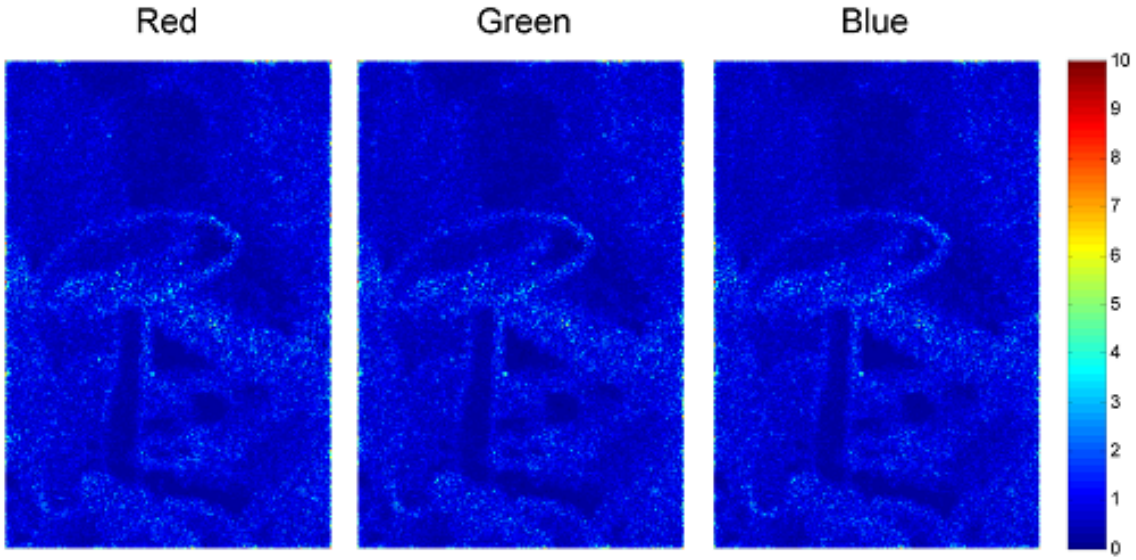


FIGURE 2.7: Expected variance of each pixel for the (Mushroom) data considered in Figure 2.6.

The truncated beta-Bernoulli process has been employed to learn dictionaries matched to image patches $\{\boldsymbol{x}_i\}_{i=1,N}$. The basic nonparametric Bayesian model is termed a beta process factor analysis (BPFA) framework, and extensions have also been considered. Specifically, the Dirichlet process (DP) has been employed to cluster the $\{\boldsymbol{x}_i\}_{i=1,N}$, encouraging similar dictionary-element usage within respective clusters. Further, the probit stick-breaking process (PSBP) has been used to impose that proximate patches are more likely to be clustered similarly (imposing that they are more probable to employ similar dictionary atoms). All inference has been performed by a Gibbs sampler, with analytic update equations. The PBFA, DP-BPFA and PSBP-BPFA have been applied to two problems in image processing: (i) denoising, and (ii) image interpolation based upon a subset of pixels selected uniformly at random. We have also considered jointly performing (i) and (ii). Important advantages of the proposed methods are: (i) a full posterior on model parameters

are inferred, and therefore "error bars" may be placed on the inverted images; (*ii*) the noise variance need not be known, it is inferred within the analysis and may be nonstationary, and it may be inferred in the presence of significant missing pixels; (*iii*) while training data may be used to initialize the dictionary learning, this is not needed, and the BPFA results are highly competitive even based upon random initializations.

The interpolation problem is related to compressive sensing (CS), in that we exploit the fact that $\{\boldsymbol{x}_i\}_{i=1,N}$ reside on a low-dimensional subspace of $\mathbb{R}^P$, such that the total number of measurements is small relative to $N \cdot P$ (recall $\boldsymbol{x}_i \in \mathbb{R}^P$). However, in CS one employs projection measurements $\boldsymbol{\Sigma}\boldsymbol{x}_i$, where $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times P}$, ideally with $n \ll P$. The interpolation problem corresponds to the special case in which the rows of $\boldsymbol{\Sigma}$ are randomly selected rows of the $P \times P$ identity matrix. This problem is closely related to the problem of matrix completion (Candès and Tao, 2010; Lawrence and Urtasun, 2009; Salakhutdinov and Mnih, 2008), where the incomplete matrix $\mathbf{X} \in \mathbb{R}^{P \times N}$ has columns defined by $\{\boldsymbol{x}_i\}_{i=1,N}$.

While the PSBP-BPFA successfully segmented the image while performing denoising and interpolation of missing pixels, we found that the PSNR performance of direct BPFA analysis performed very close to that of PSBP-BPFA in those applications. The use of PSBP-BPFA utilizes the spatial location of the image patches employed in the analysis, and therefore it removes the exchangeability assumption associated with the simple BPFA (the location of the patches may be interchanged within the BPFA, without affecting the inference). However, since in the denoising and interpolation problems we have many *overlapping* patches, the extra information provided by PSBP-BPFA does not appear to be significant. By contrast, in the CS inversion problem we do not have overlapping patches, and PSBP-BPFA provided significant performance gains relative to BPFA alone.

3

Online and Parallel Bayesian Dictionary Learning for Large Datasets

"Big Data" has recently become one of the most popular terms in machine learning field. Due to limits of memory and computation issues, it is infeasible to process the entire massive data collections at one time. There are two options to tackle this problem. One alternative way is to develop online learning technique that each time analyzes only small subset of the entire huge dataset which arrives in a stream and then discard it after one look. Such technique has been successfully applied in topic modeling (Hoffman et al., 2010; Wang et al., 2011). The other way is to develop parallel framework using Map-Reduce framework that includes a *map* step that processes subsets of data represented by key/value pairs to generate intermediate key/value pairs of sufficient statistics, and a *reduce* step that merges all intermediate values assigned with the same key to update model parameters. This framework enables to scale to massive datasets (Dean and Ghemawat, 2008).

3.1 Introduction

Dictionary learning and sparse coding are often jointly employed for many applications, such as image denoising (Elad and Aharon, 2006) and interpolation/inpainting (Mairal et al., 2008c). Numerous methods are available, for instance (Elad and Aharon, 2006; Lee et al., 2007; Mairal et al., 2008c; Zhou et al., 2012, 2009). However, the majority of these methods are based on *batch* learning, *i.e.*, they require the entire data to be loaded into memory, and hence do not scale well to very large datasets of, say, millions of signals. Such datasets require *online* learning, in which portions the data are processed sequentially.

Recent work (Mairal et al., 2010a,b) has shown the ability of online methods to successfully perform dictionary learning and sparse coding for very large images via an optimization-based approach which does not learn a statistical model for the signals. For the application to large-scale image processing, they first employ their proposed online algorithm to learn a dictionary only from large amount of *undamaged* image patches. After the dictionary has been learned, they further implement the standard sparse coding technique for inpainting. In our work, we propose a statistical model based on beta process factor analysis (BPFA) and associated online variational Bayesian (VB) (Attias, 2000) inference algorithm, which yields posterior distributions rather than point estimates for the dictionary and signals. The BPFA is related to the beta process (BP) Paisley and Carin (2009b); Thibaux and Jordan (2007c); Zhou et al. (2012, 2009), which is a nonparametric Bayesian model allowing automatic determination of the number of useful dictionary atoms. Unlike separately online learning a dictionary from *undamaged* patches and then offline performing sparse coding in (Mairal et al., 2010a,b), our online learning algorithm could simultaneously update the dictionary and implement sparse coding on the large-scale *damaged* patches within one shot.

Similar to (Hoffman et al., 2010; Wang et al., 2011), who learn topic models for very large text collections with online VB inference, we sequentially update the posterior parameters by natural gradient descent. Under appropriate choices for the learning rate, this approach is guaranteed to converge (Sato, 2001). Our online BPFA algorithm is able to efficiently compute posterior distributions for datasets of tens of millions of signals, which would be infeasible using batch inference methods, *e.g.*, Gibbs sampling as in (Zhou et al., 2012, 2009). This is demonstrated by inpainting (*i.e.*, filling missing pixels in) natural images with 12 Mpixels, similarly to (Mairal et al., 2010b).

The remainder of the paper is organized as follows. Section 2 briefly reviews the BPFA model applied to images, Section 3 describes our proposed online VB inference algorithm, and Section 4 contains experimental results. Section 5 discusses a parallel BPFA model framework using Map-Reduce paradigm that scales to massive datasets. In Section 6 a discussion of the results and final comments are concluded.

3.2 BPFA for dictionary learning in images

Chapter 2 has already introduced one Bayesian dictionary learning method with Beta Process prior, which here we will revisit for convenience. The problem of dictionary learning for the dataset $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1,\ldots,N}$ can be cast as factor analysis with $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, where $\mathbf{D} \in \mathbb{R}^{P\times K}$ is a dictionary with K atoms (factor loadings in a factor-analysis formulation), $\boldsymbol{w}_i \in \mathbb{R}^K$ are vectors of coefficients (factor scores) and $\boldsymbol{\epsilon}_i$ is a residual term that encompasses noise and deviation from the linear factor model; this deviation is assumed small. As an example, we may process images in blocks (patches) of size $B \times B$, represented as vectors $\boldsymbol{x}_i \in \mathbb{R}^P$, where $P = B^2$ is the number of pixels in each patch, and $i = 1, \ldots, N$ with N equal to the number of patches.

Following the original BPFA model (Zhou et al., 2012, 2009), it is assumed that

the vectors $\boldsymbol{w}_i$ are sparse. This is enforced by placing a beta-Bernoulli prior on $\boldsymbol{w}_i$. Specifically, define variables $\boldsymbol{z}_i \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_k)$ and $\boldsymbol{\pi} \sim \prod_{k=1}^{K} \text{Beta}(a_0/K, b_0(K-1)/K)$ where π_k is the k-th component of $\boldsymbol{\pi}$. It can be shown that this construction favors sparse binary vectors $\boldsymbol{z}_i$, as K grows large. The role of $\boldsymbol{z}_i$ is to select a subset of the columns of $\mathbf{D}$ for representing $\boldsymbol{x}_i$. The full model is

$$
\begin{aligned}
&\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i && \boldsymbol{w}_i = \boldsymbol{z}_i \odot \boldsymbol{s}_i \\
&\boldsymbol{d}_k \sim \mathcal{N}(0, P^{-1}\mathbf{I}_P) && \boldsymbol{s}_i \sim \mathcal{N}(0, \gamma_s^{-1}\mathbf{I}_K) \\
&\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P) && \pi_k \sim \text{Beta}(\frac{a_0}{K}, \frac{b_0(K-1)}{K}) \\
&\gamma_s \sim \text{Gamma}(c_0, d_0) && \gamma_\epsilon \sim \text{Gamma}(e_0, f_0)
\end{aligned}
\tag{3.1}
$$

where $\boldsymbol{d}_k$ represents the kth column (atom) of $\mathbf{D}$, $\odot$ represents the elementwise or Hadamard vector product, $\mathbf{I}_P$ ($\mathbf{I}_K$) represents a $P \times P$ ($K \times K$) identity matrix, and $\{\boldsymbol{z}_i\}_{i=1,N}$ are drawn as described above. The priors are independent for all i and k, and each $\boldsymbol{s}_i$ has its own precision γ_s. The constants $\{a_0, b_0, c_0, d_0, e_0, f_0\} = \boldsymbol{\Gamma}$ are hyperparameters, which we collect in vector $\boldsymbol{\Gamma}$. The construction in (3.1), with the beta-Bernoulli prior for $\{\boldsymbol{z}_i\}_{i=1,\ldots,N}$, is henceforth referred to as the beta process factor analysis (BPFA) model. Inference can be performed using variational Bayes (VB) or Markov chain Monte Carlo methods such as Gibbs sampling. Typically, batch implementations (as in Zhou et al. (2012)) are used; however, as discussed above, batch algorithms do not scale to very large datasets. Below, we describe an online VB method which obviates this problem.

3.3 Batch variational Bayes for BPFA

Let $\mathcal{X}$ represent observations and $\boldsymbol{\Theta}$ be model parameters. The prior on the model parameters is represented as $p(\boldsymbol{\Theta}|\boldsymbol{\Gamma})$ where $\boldsymbol{\Gamma}$ represent hyper-parameters, and the likelihood of the data is denoted as $p(\mathcal{X}|\boldsymbol{\Theta}, \boldsymbol{\Gamma})$. The goal is to infer the posterior

distribution:

$$p(\boldsymbol{\Theta}|\mathcal{X},\boldsymbol{\Gamma}) = \frac{p(\mathcal{X}|\boldsymbol{\Theta},\boldsymbol{\Gamma})p(\boldsymbol{\Theta}|\boldsymbol{\Gamma})}{\int p(\mathcal{X}|\boldsymbol{\Theta},\boldsymbol{\Gamma})p(\boldsymbol{\Theta}|\boldsymbol{\Gamma})d\boldsymbol{\Theta}} \tag{3.2}$$

where the denominator is marginal distribution, and this integral does not usually have an analytical form. Thus Variational Bayes (VB) approach proposees a variational distribution $q(\boldsymbol{\Theta})$ to approximate the true posterior distribution $p(\boldsymbol{\Theta}|\mathcal{X})$. Based on Theorem 2.1 in (M.J.Beal, 2003), a lower bound on the model marginal likelihood is

$$\mathcal{F}(q(\boldsymbol{\Theta})) = \int d\boldsymbol{\Theta} q(\boldsymbol{\Theta}) \ln \frac{p(\mathcal{X}|\boldsymbol{\Theta},\boldsymbol{\Gamma})p(\boldsymbol{\Theta}|\boldsymbol{\Gamma})}{q(\boldsymbol{\Theta})} \tag{3.3}$$

Thus maximizing $\mathcal{F}(q(\boldsymbol{\Theta}))$ simply equals to minimizing KL divergence between $q(\boldsymbol{\Theta})$ and the true posterior distribution $p(\boldsymbol{\Theta}|\mathcal{X},\boldsymbol{\Gamma})$:

$$\log p(\mathcal{X}) - \mathcal{F}(q(\boldsymbol{\Theta})) = \int d\boldsymbol{\Theta} q(\boldsymbol{\Theta}) \ln \frac{q(\boldsymbol{\Theta})}{p(\boldsymbol{\Theta}|\mathcal{X},\boldsymbol{\Gamma})} = \mathrm{KL}[q(\boldsymbol{\Theta})||p(\boldsymbol{\Theta}|\mathcal{X},\boldsymbol{\Gamma})] \geq 0$$

Also $q(\boldsymbol{\Theta})$ is assumed to be factorized, usually with the same form as employed in $p(\boldsymbol{\Theta}|\mathcal{X},\boldsymbol{\Gamma})$. With such an assumption, the variational distributions can be iteratively updated to increase the lower bound, thus to approximate the true posterior distributions.

To be specific, for the aforementioned BPFA model, VB inference is summarized as below. Given training data $\mathcal{X}$ and hyperparameters $\boldsymbol{\Gamma}$, the goal is to approximate the true posterior $p(\mathbf{D},\boldsymbol{z},\boldsymbol{s},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon|\mathcal{X},\boldsymbol{\Gamma})$ by members of a tractable class of distributions. In our case, this is chosen to be the class of fully factorized distributions of the form $q(\mathbf{D},\boldsymbol{z},\boldsymbol{s},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon) = q(\mathbf{D})q(\boldsymbol{z})q(\boldsymbol{s})q(\boldsymbol{\pi})q(\gamma_s)q(\gamma_\epsilon)$. Denoting the latent variables we wish to infer as $\mathcal{H} = \{\mathbf{D},\boldsymbol{z},\boldsymbol{s},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon\}$, VB inference maximizes the Evidence Lower BOund (ELBO):

$$\log p(\mathcal{X}|\boldsymbol{\Gamma}) \geq \mathcal{L}(\mathcal{X},\mathcal{H}) = \mathbb{E}_q[\log p(\boldsymbol{x},\mathbf{D},\boldsymbol{s},\boldsymbol{z},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon)] - \mathbb{E}_q[\log q(\mathbf{D},\boldsymbol{s},\boldsymbol{z},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon)] \tag{3.4}$$

with respect to $q(\mathcal{H})$. Maximizing the ELBO is equivalent to minimizing the Kullback-Leibler (KL) divergence between $q(\mathcal{H})$ and the true posterior $p(\mathcal{H}|\mathcal{X}, \mathbf{\Gamma})$. Following (Paisley and Carin, 2009b), we choose a fully factorized distribution q of the form

$$q(\boldsymbol{d}_k) = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad q(s_{ik}) = \mathcal{N}(\nu_{ik}, \Omega_{ik})$$

$$q(z_{ik}) = \text{Bernoulli}(\eta_{ik}) \qquad q(\pi_k) = \text{Beta}(\tau_{1k}, \tau_{2k})$$

$$q(\gamma_s) = \text{Gamma}(c', d') \qquad q(\gamma_\epsilon) = \text{Gamma}(e', f').$$

Then we could write the lower bound 3.4 as

$$\begin{aligned}\mathcal{L}(\mathcal{X}, \mathcal{H}) = &\sum_i \{\mathbb{E}_q[\log p(\boldsymbol{x}_i|\mathbf{D}, \boldsymbol{z}_i, \boldsymbol{s}_i, \gamma_\epsilon)] \\ &+ \sum_{k=1}^{K} \{\mathbb{E}_q[\log p(s_{ik}|\gamma_s)] - \mathbb{E}_q[\log q(s_{ik})] + \mathbb{E}_q[\log p(z_{ik}|\pi_k)] - \mathbb{E}_q[\log q(z_{ik})]\} \\ &+ \sum_k \{\mathbb{E}_q[\log p(\boldsymbol{d}_k|0, \frac{1}{P}\mathbf{I}_P)] - \mathbb{E}_q[\log q(\boldsymbol{d}_k)] + \mathbb{E}_q[\log p(\pi_k|a_0, b_0)] - \mathbb{E}_q[\log q(\pi_k)]\} \\ &+ \big(\mathbb{E}_q[\log p(\gamma_s|c_0, d_0)] - \mathbb{E}_q[\log q(\gamma_s)] + \mathbb{E}_q[\log p(\gamma_\epsilon|e_0, f_0)] - \mathbb{E}_q[\log q(\gamma_\epsilon)]\big)\}\end{aligned} \tag{3.5}$$

$\mathcal{L}$ can be optimized by iteratively taking derivatives with regard to parameters in $q(.)$, and setting this lower bound to be zero while fixing all other terms. The lower bound will keep increasing over iterations until the model converges. The expectations involved are taken under the variational distribution q:

$$\mathbb{E}_q[\log \gamma_s] = \psi(c') - \log d', \qquad \mathbb{E}_q[\log \gamma_\epsilon] = \psi(e') - \log f' \tag{3.6}$$

$$\mathbb{E}_q[\ln(\pi_k)] = \psi(\frac{a_0}{K} + \langle n_k \rangle) - \psi(\frac{a_0 + b_0(K-1)}{K} + N)$$

$$\mathbb{E}_q[\ln(1-\pi_k)] = \psi(\frac{b_0(K-1)}{K} + N - \langle n_k \rangle) - \psi(\frac{a_0 + b_0(K-1)}{K} + N)$$

$$\mathbb{E}_q[s_{tik}^2] = \nu_{tik}^2 + \Omega_{tik}, \qquad \mathbb{E}_q[\boldsymbol{d}_k^T \boldsymbol{d}_k] = \boldsymbol{\mu}_k^T \boldsymbol{\mu}_k + \boldsymbol{\Sigma}_k$$

$$\mathbb{E}_q[(\boldsymbol{s}_i \odot \boldsymbol{z}_i)\mathbf{D}^T\mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i)^T] = \sum_{k=1}^{K}\sum_{k \neq k'} \nu_{ik}\eta_{ik}\nu_{ik'}\eta_{ik'}\boldsymbol{\mu}_k^T\boldsymbol{\mu}_{k'} + \sum_{k=1}^{K}\eta_{ik}\mathbb{E}_q[s_{ik}^2]\mathbb{E}_q[\boldsymbol{d}_k^T\boldsymbol{d}_{k'}]$$

where $\langle n_k \rangle = \sum_{i=1}^{N} \eta_{ik}$ and $\psi(.)$ represents the digamma function. Maximizing $\ell(\cdot)$ for minibatch t, with respect to the parameters of q, leads to the following estimates. For $q(\boldsymbol{d}_k)$

$$\boldsymbol{\Sigma}_k = (P\mathbf{I}_P + \gamma_\epsilon \sum_{i=1}^{N} \mathbb{E}_q[s_{ik}^2]\eta_{ik})^{-1} \tag{3.7}$$

$$\boldsymbol{\mu}_k = \gamma_\epsilon \boldsymbol{\Sigma}_k \sum_{i=1}^{N} \nu_{ik}\eta_{ik}\mathbb{E}_q[X_i^{-k}],$$

where X_i^{-k} is the reconstruction of the i-th vector using all but the k–th atom. For $q(s_{ik})$,

$$\Omega_{ik} = (\gamma_s + \gamma_\epsilon \mathbb{E}_q[\boldsymbol{d}_k^T \boldsymbol{d}_k]\eta_{ik})^{-1} \tag{3.8}$$

$$\nu_{ik} = \gamma_\epsilon \Omega_{ik} \boldsymbol{\mu}_k^T \eta_{ik} \mathbb{E}_q[X_i^{-k}].$$

The estimate for $q(z_{ik})$ is

$$q(z_{ik}=1) \propto \exp\big[\mathbb{E}_q[\log \pi_k]\big) \times \exp\big(-\frac{\gamma_\epsilon}{2}\{\mathbb{E}_q[s_{ik}^2]\mathbb{E}_q[\boldsymbol{d}_k^T\boldsymbol{d}_k] - 2\boldsymbol{\mu}_k^T\nu_{ik}\mathbb{E}_q[X_i^{-k}]\}\big)$$

$$q(z_{ik}=0) \propto \exp\big(\mathbb{E}_q[1-\log \pi_k]\big),$$

and evaluating the expected value yields

$$\mathbb{E}_q[z_{ik}] = \eta_{ik} = \frac{q(z_{ik}=1)}{q(z_{ik}=1)+q(z_{ik}=0)}. \tag{3.9}$$

We let $z_{ik}=1$ if $\eta_{ik} \geq 0.5$ and $z_{ik}=0$ otherwise. For $q(\boldsymbol{\pi})$ the estimates are

$$\tau_{1k} = \frac{a_0}{K} + \langle n_k \rangle, \qquad \tau_{2k} = \frac{b_0(K-1)}{K} + N - \langle n_k \rangle, \tag{3.10}$$

and for $q(\gamma_s)$ and $q(\gamma_\epsilon)$ we have

$$c' = c_0 + \frac{NK}{2} \qquad d' = d_0 + \frac{1}{2}\sum_{i=1}^{N} \boldsymbol{\nu}_i^T \boldsymbol{\nu}_i \qquad e' = e_0 + \frac{NP}{2}$$

$$f' = f_0 + \frac{1}{2}\sum_{i=1}^{N}\{\boldsymbol{x}_i\boldsymbol{x}_i^T - 2\sum_{k=1}^{K}\eta_{ik}\nu_{ik}\boldsymbol{\mu}_k^T\boldsymbol{x}_i + \mathbb{E}_q[(\boldsymbol{s}_i \odot \boldsymbol{z}_i)\mathbf{D}^T\mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i)^T]\}$$

Algorithm 1 outlines batch VB for BPFA.

Algorithm 1 Batch variational Bayes for Dictionary Learning

Initialize $\mathcal{H}$ randomly
while Stopping criterion is not met **do**
    for $k = 1$ to K **do**
        Estimate $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$
        For $i = 1, \ldots, N$: Estimate η_{ik}, Ω_{ik} and ν_{ik}
        Estimate τ_{1k} and τ_{2k}
    end for
    Update c', d', e', f'
end while

3.4 Online variational Bayes for BPFA

Before further specifying how to optimize $q(\mathcal{H})$, we describe how to adapt the standard VB formulation to the online setting. We carry out online learning by stochastic coordinate ascent, in a manner analogous to Hoffman et al. (2010). The dataset $\mathcal{X}$ is randomly partitioned into T subsets (called *minibatches*) of vectors with N_t vectors per minibatch t. We use constant N_t for all t. In the limit, we can process one patch at a time by setting $T = N$ and $N_t = 1$, or revert to a batch algorithm by setting $T = 1$ and $N_t = N$. The advantages of this procedure are two-fold: (i) it is only necessary to store N_t vectors at a time in memory, independently of N; (ii) by processing the data in a random sequence, we gain robustness (albeit not immunity) to local optima and maintain convergence guarantees Sato (2001). We now write $q(\mathcal{H})$ taking into account the minibatch index t:

$$
\begin{aligned}
q(\boldsymbol{d}_k) &= \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) & q(s_{tik}) &= \mathcal{N}(\nu_{tik}, \Omega_{tik}) \\
q(z_{tik}) &= \text{Bernoulli}(\eta_{tik}) & q(\pi_k) &= \text{Beta}(\tau_{1k}, \tau_{2k}) \\
q(\gamma_s) &= \text{Gamma}(c', d') & q(\gamma_\epsilon) &= \text{Gamma}(e', f').
\end{aligned}
\tag{3.11}
$$

In the above distributions, $s_{tik}, \nu_{tik}, \Omega_{tik}, z_{tik}$ and η_{tik} all refer to the i–th patch in minibatch t and to the k-th dictionary atom. The next step is to derive estimates for the posterior parameters $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \nu_{tik}, \Omega_{tik}, \eta_{tik}, \tau_{1k}, \tau_{2k}, c', d', e', f'$. The bound (3.4)

expands to

$$\begin{aligned}
\mathcal{L}(\mathcal{X},\mathcal{H}) &= \sum_t \sum_i \{\mathbb{E}_q[\log p(\boldsymbol{x}_{ti}|\mathbf{D},\boldsymbol{z}_{ti},\boldsymbol{s}_{ti},\gamma_\epsilon)] \\
&+ \sum_{k=1}^{K}\{\mathbb{E}_q[\log p(s_{tik}|\gamma_s)] - \mathbb{E}_q[\log q(s_{tik})] + \mathbb{E}_q[\log p(z_{tik}|\pi_k)] - \mathbb{E}_q[\log q(z_{tik})]\} \\
&+ \sum_k \{\mathbb{E}_q[\log p(\boldsymbol{d}_k|0,\frac{1}{P}\mathbf{I}_P)] - \mathbb{E}_q[\log q(\boldsymbol{d}_k)] + \mathbb{E}_q[\log p(\pi_k|a_0,b_0)] - \mathbb{E}_q[\log q(\pi_k)]\}/T \\
&+ \big(\mathbb{E}_q[\log p(\gamma_s|c_0,d_0)] - \mathbb{E}_q[\log q(\gamma_s)] + \mathbb{E}_q[\log p(\gamma_\epsilon|e_0,f_0)] - \mathbb{E}_q[\log q(\gamma_\epsilon)]\big)/T\} \\
&= \sum_t \ell(\boldsymbol{x}_t,\boldsymbol{s}_t,\boldsymbol{z}_t,\mathbf{D},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon)
\end{aligned} \tag{3.12}$$

where $\ell(\boldsymbol{x}_t,\boldsymbol{s}_t,\boldsymbol{z}_t,\mathbf{D},\boldsymbol{\pi},\gamma_s,\gamma_\epsilon)$ denotes the contribution of minibatch t to the ELBO. As in Hoffman et al. (2010), the ELBO is written as a sum over t, thereby enabling an online inference method. The expectations involved are taken under the variational distribution q:

$$\mathbb{E}_q[\log\gamma_s] = \psi(c') - \log d', \qquad \mathbb{E}_q[\log\gamma_\epsilon] = \psi(e') - \log f' \tag{3.13}$$

$$\mathbb{E}_q[\ln(\pi_k)] = \psi(\frac{a_0}{K} + \frac{N}{N_t}\langle n_{tk}\rangle) - \psi(\frac{a_0 + b_0(K-1)}{K} + N)$$

$$\mathbb{E}_q[\ln(1-\pi_k)] = \psi(\frac{b_0(K-1)}{K} + N - \frac{N}{N_t}\langle n_{tk}\rangle) - \psi(\frac{a_0 + b_0(K-1)}{K} + N)$$

$$\mathbb{E}_q[s_{tik}^2] = \nu_{tik}^2 + \Omega_{tik}, \qquad \mathbb{E}_q[\boldsymbol{d}_k^T\boldsymbol{d}_k] = \boldsymbol{\mu}_k^T\boldsymbol{\mu}_k + \boldsymbol{\Sigma}_k$$

$$\mathbb{E}_q[(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})\mathbf{D}^T\mathbf{D}(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})^T] = \sum_{k=1}^{K}\sum_{k\neq k'}\nu_{tik}\eta_{tik}\nu_{tik'}\eta_{tik'}\boldsymbol{\mu}_k^T\boldsymbol{\mu}_{k'} + \sum_{k=1}^{K}\eta_{tik}\mathbb{E}_q[s_{tik}^2]\mathbb{E}_q[\boldsymbol{d}_k^T\boldsymbol{d}_{k'}]$$

where $\langle n_{tk}\rangle = \sum_{i=1}^{N_t}\eta_{tik}$ and $\psi(.)$ represents the digamma function. Maximizing $\ell(\cdot)$ for minibatch t, with respect to the parameters of q, leads to the following estimates.

3.4.1 Per-minibatch parameter estimates

For $q(\boldsymbol{d}_k)$,

$$\boldsymbol{\Sigma}_k = (P\mathbf{I}_P + \gamma_\epsilon \frac{N}{N_t} \sum_{i=1}^{N_t} \mathbb{E}_q[s_{tik}^2]\eta_{tik})^{-1} \tag{3.14}$$

$$\boldsymbol{\mu}_k = \gamma_\epsilon \boldsymbol{\Sigma}_k \frac{N}{N_t} \sum_{i=1}^{N_t} \nu_{tik}\eta_{tik}\mathbb{E}_q[X_{ti}^{-k}],$$

where X_{ti}^{-k} is the reconstruction of the i-th vector in minibatch t using all but the k–th atom. For $q(s_{itk})$,

$$\Omega_{tik} = (\gamma_s + \gamma_\epsilon \mathbb{E}_q[\boldsymbol{d}_k^T \boldsymbol{d}_k]\eta_{tik})^{-1} \tag{3.15}$$

$$\nu_{tik} = \gamma_\epsilon \Omega_{tik} \boldsymbol{\mu}_k^T \eta_{tik} \mathbb{E}_q[X_{ti}^{-k}].$$

The estimate for $q(z_{tik})$ is

$$q(z_{tik}=1) \propto \exp\big[\mathbb{E}_q[\log \pi_k]\big) \times \exp\big(-\frac{\gamma_\epsilon}{2}\{\mathbb{E}_q[s_{tik}^2]\mathbb{E}_q[\boldsymbol{d}_k^T\boldsymbol{d}_k] - 2\boldsymbol{\mu}_k^T\nu_{tik}\mathbb{E}_q[X_{ti}^{-k}]\}\big)$$

$$q(z_{tik}=0) \propto \exp\big(\mathbb{E}_q[1-\log \pi_k]\big),$$

and evaluating the expected value yields

$$\mathbb{E}_q[z_{tik}] = \eta_{tik} = \frac{q(z_{tik}=1)}{q(z_{tik}=1)+q(z_{tik}=0)}. \tag{3.16}$$

We let $z_{tik}=1$ if $\eta_{tik} \geq 0.5$ and $z_{tik}=0$ otherwise. For $q(\boldsymbol{\pi})$ the estimates are

$$\tau_{1k} = \frac{a_0}{K} + \frac{N}{N_t}\langle n_{tk}\rangle, \qquad \tau_{2k} = \frac{b_0(K-1)}{K} + N - \frac{N}{N_t}\langle n_{tk}\rangle, \tag{3.17}$$

and for $q(\gamma_s)$ and $q(\gamma_\epsilon)$ we have

$$c' = c_0 + \frac{NK}{2} \qquad d' = d_0 + \frac{N}{2N_t}\sum_{i=1}^{N_t} \boldsymbol{\nu}_{ti}^T\boldsymbol{\nu}_{ti} \qquad e' = e_0 + \frac{NP}{2}$$

$$f' = f_0 + \frac{N}{2N_t}\sum_{i=1}^{N_t}\{\boldsymbol{x}_{ti}\boldsymbol{x}_{ti}^T - 2\sum_{k=1}^{K}\eta_{tik}\nu_{tik}\boldsymbol{\mu}_k^T\boldsymbol{x}_{ti} + \mathbb{E}_q[(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})\mathbf{D}^T\mathbf{D}(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})^T]\}$$

3.4.2 Global parameter updates

Let $\mathbf{\Lambda} = \{\boldsymbol{\mu}_k, \mathbf{\Sigma}_k, \tau_{1k}, \tau_{2k}, c', d', e', f'\}$ be a vector containing the parameters of q that are not specific to the minibatch and data indices t and i. Define also $\tilde{\mathbf{\Lambda}}$ as the estimates of $\mathbf{\Lambda}$ based solely on the most recent minibatch, as computed in Section 3.1. We call $\mathbf{\Lambda}$ the *global* parameters. These are sequentially updated by computing a weighted average between their previous value and $\tilde{\mathbf{\Lambda}}$:

$$\mathbf{\Lambda} \leftarrow (1 - \rho_t)\mathbf{\Lambda} + \rho_t \tilde{\mathbf{\Lambda}}. \tag{3.18}$$

where ρ_t is the weight given to each new minibatch. Following Hoffman et al. (2010), in the above estimates the contribution of each datum i is weighted by N/N_t, so as to make $\tilde{\mathbf{\Lambda}}$ equal to the batch parameter estimate we would obtain if the latest minibatch constituted the entirety of the dataset. The weight ρ_t, also called the *learning step*, is chosen according to the schedule $\rho_t = (\tau_0 + t)^{-\kappa}$, where $\kappa \in (0.5, 1]$ controls the rate of decay of the contribution from old minibatches and $\tau_0 \geq 0$ serves to slow down the initial iterations. It is shown in Hoffman et al. (2010); Sato (2001) that this procedure converges and is equivalent to stochastic *natural gradient* ascent. The natural gradient is obtained by multiplying the standard gradient of the objective function by an appropriate metric matrix (in our case, the inverse Fisher information matrix for q). The overall method is summarized in Algorithm 2.

3.4.3 Convergence Analysis

From the expression 3.12, we can see that the lower bound $\mathcal{L}(\mathcal{X}, \mathcal{H})$ can be written as

$$\mathcal{L}(\mathcal{X}, \mathcal{H}) = T\mathbb{E}[\mathcal{L}_t(\mathcal{X}_t, \mathcal{H}|\mathbf{\Lambda})] \tag{3.19}$$

where $\mathcal{L}_t$ is defined as in equation 3.5 and $\mathcal{X}_t$ denotes the data in the minibatch t. The expectation is taken over the empirical distribution of the data set as (Hoffman et al., 2010). The expression $T\mathcal{L}_t(\mathcal{X}_t, \mathcal{H}|\mathbf{\Lambda})$ is the variational lower bound evaluated

Algorithm 2 Online variational Bayes for Dictionary Learning

Define $\rho_t = (\tau_0 + t)^{-\kappa}$
Initialize $\mathbf{\Lambda}^{(1)}$ by MCMC on a small random subset of $\mathbf{X}$
for $t = 2$ to ∞ **do**
    while Stopping criterion is not met **do**
        for $k = 1$ to K **do**
            Estimate $\boldsymbol{\mu}_k$ and $\mathbf{\Sigma}_k$
            For $i = 1, \ldots, N_t$: Estimate η_{tik}, Ω_{tik} and ν_{tik}
            Estimate τ_{1k} and τ_{2k}
        end for
        Update c', d', e', f'
    end while
    Compute $\tilde{\mathbf{\Lambda}}$
    $\mathbf{\Lambda}^{(t)} = (1 - \rho_t)\mathbf{\Lambda}^{(t-1)} + \rho_t \tilde{\mathbf{\Lambda}}$
end for

with T duplicated copies of minibatch t. We can optimize equation 3.19 over $\mathbf{\Lambda}$ by to obtain the stochastic approximation for finding the maximum of the expected KL divergence

$$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda} + \rho_t T \bigtriangledown_{\mathbf{\Lambda}} \mathcal{L}_t(\mathcal{X}_t, \mathcal{H}) \tag{3.20}$$

where the learning rate ρ_t should satisfy the condition (Bottou, 1998)

$$\sum_{t=1}^{\infty} \rho_t = \infty, \quad \text{and} \quad \sum_{t=1}^{\infty} \rho_t^2 \leq \infty \tag{3.21}$$

to ensure that both $\mathbf{\Lambda}$ converges and $\rho_t \bigtriangledown_{\mathbf{\Lambda}} \sum_t \mathcal{L}_t(\mathcal{X}_t, \mathcal{H})$ converges to 0, and thus that $\mathbf{\Lambda}$ converges to a stationary point.

Since the gradient of the variational objective contains the covariance matrix of the variational distribution, the update in equation 3.20 only consider first-order gradient information. Natural gradient method is employed to speed up the online inference by multiplying the gradient by the inverse of the Riemannian metric (Amari, 1998). The covariance matrix of the variational distribution is then cancelled when multiplying the gradient by the inverse of Riemannian metric, which makes the natural gradient easy and fast to converge. (Hoffman et al., 2010; Wang et al., 2011) shows to implement online VB algorithm by such efficient stochastic natural gradient descent method.

3.5 Parallel variational Bayes for BPFA

In this section, we develop a parallel BPFA framework based on Map-Reduce that fits the model in a scalable way (Dean and Ghemawat, 2008). Similar as the online learning technique, we first randomly partition the data into small partitions, and each partition is assigned to one node at the "Map" step. We run VB on each Mapper in parallel to obtain sufficient statistics of estimates of per-minibatch parameters. Then these sufficient statistics are collected from each "Mapper" to obtain the estimates of global parameters $\mathbf{\Lambda}$. This part is called "Reducer" step. The key of handling scalable data sets is to parallely update and store a small subset of per-minibatch parameters in memory on each node at the "Map" step. For our case, only one reducer is required for two reasons: (a) It is only the updates of per-minibatch parameters that are most computation expensive at the "Map" step. (b)The updates of global parameters only depends on the summation of sufficient statistics of per-minibatch parameters, which will be redundant if multiple reducers are assigned. Iterations between "Map" and "Reduce" are operated until convergence is met. Next we will provide a detailed study on how to implement Map-Reduce framework of BPFA model.

3.5.1 "Map" Step

At the "Map" step, per-minibatch parameters $\boldsymbol{s}$ and $\boldsymbol{z}$ are estimated in parallel at each node/mapper. In the beginning of each mapper, global parameters $\mathbf{\Lambda}$ as well as hyper parameters are loaded. At the first iteration, these global parameters are their initializations. From the second iteration, the global parameters are the direct output from the "Reduce" step. For each data $\boldsymbol{x}_{ti}$ assigned to mapper t where $i = 1, ..., N_t$, the updates of $\boldsymbol{s}_{ti}$ and $\boldsymbol{z}_{ti}$ follow 3.15 and 3.16. Furthermore, to prepare for the updates of global parameters at the "Reduce" step, the sufficient statistics of these

per-minibatch parameters $\{\boldsymbol{s}_{ti}, \boldsymbol{z}_{ti}\}_{i=1}^{N_t}$ also need to be computed as

$$\mathcal{SS}^t_{\boldsymbol{\Sigma}_k} = \sum_{i=1}^{N_t} \mathbb{E}_q[s^2_{tik}]\eta_{tik}), \qquad \mathcal{SS}^t_{\boldsymbol{\mu}_k} = \sum_{i=1}^{N_t} \nu_{tik}\eta_{tik}\mathbb{E}_q[X^{-k}_{ti} \tag{3.22}$$

$$\mathcal{SS}^t_{n_{tk}} = \sum_{i=1}^{N_t} z_{tik}, \qquad\qquad \mathcal{SS}^t_{s_{l2}} = \sum_{i=1}^{N_t} \boldsymbol{\nu}^T_{ti}\boldsymbol{\nu}_{ti}$$

$$\mathcal{SS}^t_{error} = \sum_{i=1}^{N_t}\{\boldsymbol{x}_{ti}\boldsymbol{x}^T_{ti} - 2\sum_{k=1}^{K}\eta_{tik}\nu_{tik}\boldsymbol{\mu}^T_k\boldsymbol{x}_{ti} + \mathbb{E}_q[(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})\mathbf{D}^T\mathbf{D}(\boldsymbol{s}_{ti}\odot\boldsymbol{z}_{ti})^T]\}$$

After these five sufficient statistics are computed, they are going to be passed to the " Reduce" step with the associated key of mapper index t denoting which mapper they belong to. In summary, per-minibatch parameters are updated and their sufficient statistics are stored at the "Map" step and passed to the "Reduce" step.

3.5.2 "Reduce" Step

At the "Reduce" step, global parameters are going to be updated. In the beginning of the reducer, sufficient statisctics 3.22 of each per-minibatch parameters are collected from all the mappers, each associated with its own key. The summation os these sufficient statistics will be used for the updates of global parameters $\boldsymbol{\Lambda} = \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \tau_{1k}, \tau_{2k}, c', d', e', f'\}$, listed as

$$\boldsymbol{\Sigma}_k = (P\mathbf{I}_P + \gamma_\epsilon\sum_t \mathcal{SS}^t_{\boldsymbol{\Sigma}_k})^{-1}, \qquad \boldsymbol{\mu}_k = \gamma_\epsilon\boldsymbol{\Sigma}_k\sum_t \mathcal{SS}^t_{\boldsymbol{\mu}_k}, \tag{3.23}$$

$$\tau_{1k} = \frac{a_0}{K} + \sum_t \mathcal{SS}^t_{n_{tk}}, \qquad \tau_{2k} = \frac{b_0(K-1)}{K} + N - \sum_t \mathcal{SS}^t_{n_{tk}},$$

$$c' = c_0 + \frac{NK}{2}, \qquad\qquad d' = d_0 + \frac{1}{2}\sum_t \mathcal{SS}^t_{s_{l2}},$$

$$e' = e_0 + \frac{NP}{2}, \qquad\qquad f' = f_0 + \frac{1}{2}\sum_t \mathcal{SS}^t_{error}$$

After the global parameters are updated, they will be passed back to the "Map" step, denoted as one iteration. Multiple iterations are required until the convergence. Therefore, the parallel framework for BPFA model is summarized in Algorithm 3

Algorithm 3 Parallel variational Bayes for Dictionary Learning

Initialize $\mathbf{\Lambda}$ and store it as a side input
Partition data into T partitions using random seed
for each partition $t \in \{1, ..., T\}$ **do**
    Initialize $\boldsymbol{s}_{ti}$, $\boldsymbol{z}_{ti}$, and $\mathbf{X}_t^{-k}$
end for
while Stopping criterion is not met **do**
    for t = 1 to T **do**
        for k = 1 to K **do**
            For $i = 1, \ldots, N_t$: Estimate $\boldsymbol{s}_{tik}$ and $\boldsymbol{z}_{tik}$ in 3.14
            Compute sufficient statistics in 3.22 and store them with key t
        end for
    end for
    Update $\mathbf{\Lambda}$ in 3.23 and replace the side input
end while

3.5.3 Discussion

While the BPFA model has successfully been applied on image inpainting and/or denoising, shown in section 3.6, it could also naturally fit the personalized item recommendation problem, which is one of the most popular topics in industry. Let $\mathbf{X} \in \mathbb{R}^{M \times N}$ be the Netflix rating matrix with a large fraction of the elements missing, and to be inferred, where N, the number of users, is much larger than M, the number of movies. However, it is even hard to load the entire data to fit into memory of a single machine. One option to solving this issue is to use distributed computing framework like Map-Reduce (Dean and Ghemawat, 2008) introduced above. One thing to be noted is that data partitioning strategy used in the Map-Reduce phase would crucially affect the model performance according to (Dean and Ghemawat, 2008), since the number of users is far more than the number of movies. Therefore we partition data by users that each minibatch include all the ratings for a fraction of users assigned to this minibatch, which guarantees that all data from a user belongs to the same partition.

3.6 Experiments

3.6.1 Parameter settings

We demonstrate the performance of our algorithm in two image inpainting tasks, where we do not observe all the pixels in each patch. Hence, for patch i we do not directly observe $\boldsymbol{x}_i$, but rather random projections $\boldsymbol{y}_i = \mathbf{A}_i\boldsymbol{x}_i$, where $\mathbf{A}_i \in \mathbb{R}^{m_i \times P}$ is a concatenation of random rows of the $P \times P$ identity matrix $\mathbf{I}_P$ and we observe m_i pixels. The BPFA statistical model is well suited to this missing data problem Zhou et al. (2012). For all experiments, we used $K = 256$ atoms and set the hyperparameters to $a_0 = b_0 = 1, c_0 = d_0 = e_0 = f_0 = 10^{-6}$, which are standard "flat" values. The online learning parameters were set to $\tau_0 = 1000$ and $\kappa = 0.5$, and we divided the dataset in $T = 10,000$ minibatches of size $N_t \approx 1200$, which we found to be a good compromise between computational efficiency and convergence stability. For the first minibatch only, we initialize the model using the Gibbs sampling algorithm in Zhou et al. (2012).

3.6.2 Castle Image

The first experiment consists of taking the well-known "castle" RGB image of size 321×481, removing 50% of the pixels uniformly at random and then reconstructing. The aim of this experiment is to show that the online VB algorithm works as well as the batch MCMC algorthim via a regular RGB image. The result is shown in Figure 1, and we obtain a peak signal-to-noise ratio (PSNR) very similar to that of the batch Gibbs sampling algorithm from Zhou et al. (2012) on the same image.

3.6.3 12 Million-pixels Image

The second experiment is a difficult inpainting task, using the same RGB "bird" image of size 3000×4000 (12 Mpixel) as in Mairal et al. (2010b). Due to the use of overlapping patches, N is slightly under 12 million. We restore the image from two

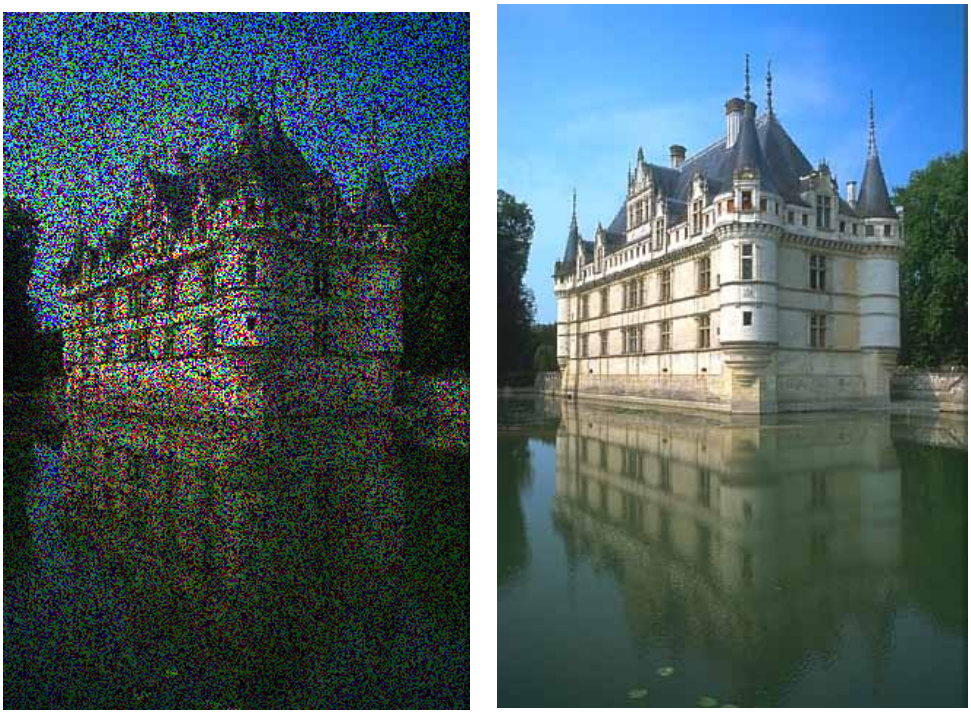


FIGURE 3.1: Inpainting example on a medium-sized image with 50% missing pixels (left). The reconstructed image (right) has PSNR=36.53 dB, virtually identical to that of a batch implementation using Gibbs sampling Zhou et al. (2012) (PSNR=36.45).

types of damage: superimposed text, and a varying percentage of missing-at-random pixels (which are set to zero). The patches are of size $8 \times 8 \times 3$. Unlike Mairal et al. (2010b), we simultaneously learn a dictionary and reconstruct the image. A non-parallelized MATLAB implementation of our Algorithm 1 takes approximately 18 hours for one full pass through the dataset (one epoch). The PSNR results can be seen in Table 1, and we illustrate the reconstruction in Figure 2, for the cases of superimposed text and 90% missing pixels. For the latter, we also show the reconstruction using nearest-neighbor (NN) interpolation with neighborhood size five. Our approach achieves better PSNR (34.53 dB vs. 16.65 dB) and avoids the multiple artifacts present in the kNN version. Note that, while Mairal et al. (2010b) do not report PSNR, their reconstruction from superimposed text is visually

Table 3.1: Peak signal-to-noise (PSNR) for the reconstruction with varying percentage of missing-at-random pixels and for text-damaged image

	90%	80%	70%	60%	50%	text
PSNR(dB)	34.53	40.24	43.31	45.24	46.72	45.73

indistinguishable from ours. The posterior mean of the learned dictionary is shown in Figure 3, and only 41 out of 256 atoms are used among all patches; individual patches use significantly less than 41 atoms. We also obtain the posterior variance, which is shown in Figure 2. Note that the pixels situated in edges and high-complexity textures have higher variance than those located in smooth regions.

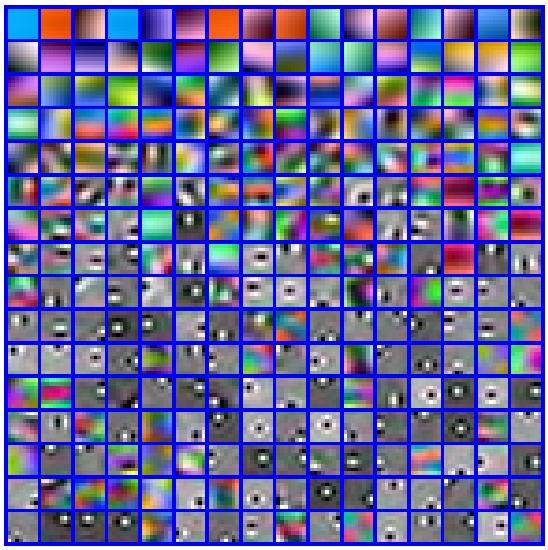


FIGURE 3.2: Posterior mean of the learned dictionary for the 12 Mpixel "bird" image. There are 256 atoms, but only 41 are used.

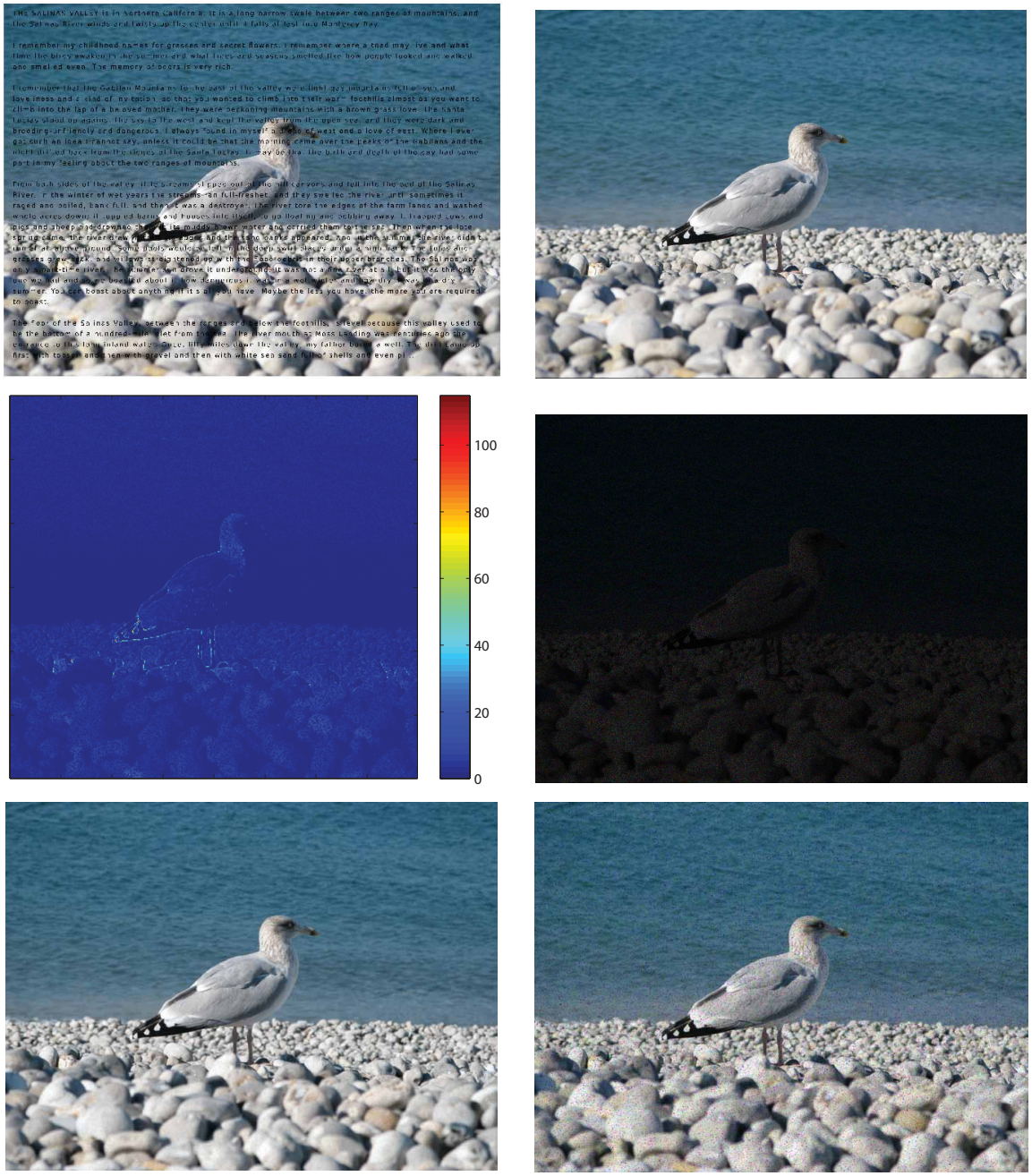


FIGURE 3.3: Inpainting example on a 12-Mpixel image. Top left: Image damaged by text; Top middle: reconstruction (PSNR=45.73 dB); Top right: variance. Bottom left: Image with 90% missing pixels (shown as black); Bottom middle: reconstruction (PSNR=34.53 dB); Bottom right: reconstruction using NN interpolation with neighborhood size five (PSNR=16.65 dB, multiple artifacts). Best viewed in color with electronic zooming.

3.7 Summary

We have presented an online variational Bayes algorithm for analysis of very large datasets, applied to image inpainting. The algorithm infers a BPFA statistical model, enjoys converge guarantees and can be interpreted as natural gradient optimization. State-of-the-art performance is achieved in inpainting problems with very large natural images. This is one of very few methods capable of simultaneously performing dictionary learning and sparse coding for images of tens of millions of pixels while also inferring the number of factors. Additionally, and unlike other methods, (approximate) full posterior distributions are learned rather than point estimates. This enables further analysis and allows other problems to be considered, such as topic modeling and active learning.

4

On the Integration of Dictionary Learning and Topic Modeling

4.1 Introduction

In this chapter we integrate feature learning and topic modeling within a unified setting. The feature extraction is performed using dictionary learning, with this integrated within topic modeling. Recent research on dictionary learning and sparse coding has demonstrated superior performance in a number of challenging image processing applications, including image denoising, inpainting and sparse image modeling (Mairal et al., 2008a; Zhou et al., 2009). Recent advances in image classification show that substantially improved performance may be achieved by extracting features from local descriptors with dictionary learning and sparse coding, this replacing VQ Yang et al. (2009). In the work reported here we also replace VQ, with the number of features (dictionary atoms) and their characteristics inferred via a new application of the hierarchical beta process Thibaux and Jordan (2007a).

We develop a novel hierarchical Bayesian model that integrates dictionary learning, sparse coding and topic modeling, for joint analysis of multiple images and (when

present) associated annotations. The model defines topics in terms of the probabilities with which dictionary atoms are used, with the dictionary learned jointly while performing topic modeling. The learned model clusters all images into groups, based upon dictionary usage; a statistical distribution is also provided for words that may be associated with previously non-annotated images (only a subset of the images are assumed annotated when learning the model). The encouraging performance of the framework is demonstrated on several commonly analyzed datasets, with comparisons to previous related research. We also quantitatively examine the utility of jointly performing image feature learning and topic modeling, *vis-a-vis* treating these as two disjoint processes. Additionally, we compare the performance of learned features applied directly to the image, as opposed to first doing feature extraction using such methods as SIFT. To the authors' knowledge, this paper is the first to unify dictionary learning and statistical topic modeling.

4.2 Model Construction

We wish to analyze M images, and a subset of the images have accompanying words or an annotation; the vocabulary of such annotations is assumed to be of dimension L. The vector $\boldsymbol{x}_m$ represents the pixels associated with image m, and $\boldsymbol{y}_m = (y_{m1}, \ldots, y_{mL})^T$ represents a vector of word counts for that image, when available (y_{ml} represents the number of times word $l \in \{1, \ldots, L\}$ is present in the annotation). The objective is to organize/sort/cluster the images, utilizing annotations when available.

The M images are assumed characterized by the following hierarchy. Each image is assumed to have an associated category/class. For example, some images may be characterized as city scenes, while others may be forest or beach scenes. The number of such categories is not set or defined *a priori*, and is to be inferred by the data under analysis. At the next level of the hierarchy, each image category is

characterized in terms of a distribution of objects/entities that may appear in the image (these image objects are analogous to topics in topic models). Again, the number of such objects is to be inferred by the data, and the partial presence of annotations plays an important role in defining an appropriate number of objects.

Finally, each object (or topic) is characterized at the patch level in terms of a distribution over dictionary atoms. The number of dictionary atoms and their composition are also inferred based on the data under test. The dictionary atoms play the role of words in topic models. In classical topic models Blei et al. (2003b) each topic is characterized by a distribution over words. In the analysis that follows, each topic is characterized by a set of probabilities, defining the probabilities with which particular dictionary atoms ("words") are selected to represent a particular object.

4.2.1 Hierarchical BP & Dictionary learning

When presenting the model we start at the level of the observed pixels, and then work our way up to the top (image-class) level. As is customary in dictionary learning applied to image analysis, we divide each image into partially overlapping patches, where each patch consists of a contiguous subset of pixels. Specifically, the mth image is divided into N_m patches, where the ith patch is denoted $\boldsymbol{x}_{mi} \in \mathbb{R}^P$ with $i = 1, \ldots, N_m$.

Each patch $\boldsymbol{x}_{mi}$ is represented as a sparse linear combination of learned dictionary atoms. Further, each patch is assumed associated with an object/entity ("topic"); the probability of which dictionary atoms are employed for a given patch is dictated by the object associated with it. The connection between the different topics, the dictionary usage, and the dictionary form is constituted via a *hierarchical* beta process (HBP) Thibaux and Jordan (2007a), in the following manner.

Each patch is represented as $\boldsymbol{x}_{mi} = \mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}) + \epsilon_{mi}$, where $\odot$ represents the

element-wise/Hadamard product, $\mathbf{D} = [\boldsymbol{d}_1, \cdots, \boldsymbol{d}_K] \in \mathbb{R}^{P \times K}$, K is the truncation level on the possible number of dictionary atoms, $\boldsymbol{z}_{mi} = [z_{mi1}, \cdots, z_{miK}]^T$, $\boldsymbol{s}_{mi} = [s_{mi1}, \cdots, s_{miK}]^T$, $z_{mik} \in \{0, 1\}$ indicates whether the kth atom is *active* within patch i in image m, $s_{mik} \in \mathbb{R}^+$, and $\boldsymbol{\epsilon}_{mi}$ is the residual error. Note that $\boldsymbol{z}_{mi}$ represents the specific sparseness pattern of dictionary usage for $\boldsymbol{x}_{mi}$. The hierarchical form of the model is

$$\begin{aligned} \boldsymbol{x}_{mi} &\sim \mathcal{N}(\mathbf{D}(\boldsymbol{z}_{mi} \circ \boldsymbol{s}_{mi}), \gamma_\epsilon^{-1}\mathbf{I}_P) \\ \boldsymbol{d}_k &\sim \mathcal{N}(0, \frac{1}{P}\mathbf{I}_P) \\ \boldsymbol{s}_{mi} &\sim \mathcal{N}_+(0, \gamma_s^{-1}\mathbf{I}_K) \\ \boldsymbol{z}_{mi} &\sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{h_{mi}k}) \end{aligned} \tag{4.1}$$

where gamma priors are placed on both γ_ϵ and γ_s. Unlike conventional dictionary learning Zhou et al. (2009), positive weights $\boldsymbol{s}_{mi}$ (truncated normal, $\mathcal{N}_+(\cdot)$) are imposed, which we have found to yield improved results.

In (5.3) the indicator variable h_{mi} defines the topic associated with $\boldsymbol{x}_{mi}$, and this will be controlled via higher layers of the model; we discuss this below. We now focus on how the probabilities π_{hk} are constituted, in terms of an HBP. Specifically, the K-dimensional vector $\boldsymbol{\pi}_h$ defines the probability that each of the K columns of $\mathbf{D}$ is employed to represent object type $h \in \{1, \ldots, J\}$, where the kth component of $\boldsymbol{\pi}_h$ is π_{hk}. Using an HBP construction as in Thibaux and Jordan (2007a), these probability vectors are defined as

$$\boldsymbol{\pi}_h \sim \prod_{k=1}^{K} \text{Beta}(c_1\eta_k, c_1(1-\eta_k)) \ , \ \eta_k \sim \text{Beta}(c_0\eta_0, c_0(1-\eta_0)) \tag{4.2}$$

where η_k represents the "global" probability of using dictionary atom $\boldsymbol{d}_k$ across all topics (object types), and π_{hk} represents the probability of using $\boldsymbol{d}_k$ for object type

h. Although the model is truncated to J topics, in practice J is set to a large value, and the model infers which subset of $\{\boldsymbol{\pi}_h\}$ are actually needed to represent the observed data. Similarly, K is set to a large value, and the model infers the subset of dictionary atoms ("words") needed to represent the data.

In the Indian buffet metaphor Griffiths and Ghahramani (2005a); Thibaux and Jordan (2007a), each of the topics is a customer at a buffet of dictionary atoms ("words" in the context of a topic model). The vector $\boldsymbol{\pi}_h$ defines the probability of dictionary atom selection for topic/customer h. While each topic shares the same buffet of dictionary atoms, the probability with which such are selected is topic-dependent.

4.2.2 Topic-modeling component

The generative model has now constituted a set of topic-dependent dictionary-usage probabilities $\{\boldsymbol{\pi}_h\}$, and a given image patch $\boldsymbol{x}_{mi}$ is linked to an indicator variable $h_{mi} \in \{1, \ldots, J\}$ defining the topic associated with patch i in image m. What remains is to define probabilities with which objects/topics may be found in an image, and to link this probability vector to the specific image class under test.

Let $r_m \in \{1, \ldots, T\}$ represent the image class associated with image m, which we seek to cluster. Then the remainder of the generative process may be expressed as

$$h_{mi} \sim \sum_{j=1}^{J} \nu_{r_m j} \delta_j \ , \ \boldsymbol{\nu}_t \sim \mathrm{Dir}(\alpha_\nu/J, \cdots, \alpha_\nu/J)$$

$$r_m \sim \sum_{t=1}^{T} \mu_t \delta_t \ , \ \boldsymbol{\mu} \sim \mathrm{Dir}(\alpha_\mu/T, ..., \alpha_\mu/T) \tag{4.3}$$

where δ_α is a unit measure at the point α. The J-dimensional probability vector $\boldsymbol{\nu}_t$ defines the probability with which each of the J objects are manifested in image

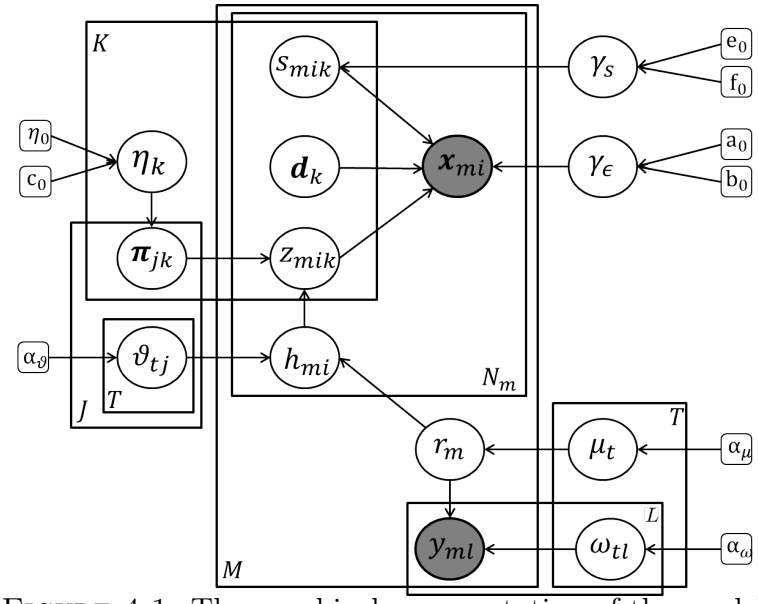


FIGURE 4.1: The graphical representation of the model.

class t, while $\boldsymbol{\mu}$ defines the probability with which the T image classes are manifested across the M images.

Summarizing the generative process thus far, for image m we draw a latent $r_m \in \{1, \ldots, T\}$, this defining the image class. For each of the image patches $\{\boldsymbol{x}_{mi}\}$ in this image we draw an associated object type or topic, with probability of topics defined by $\boldsymbol{\nu}_{r_m}$. Latent $h_{mi} \in \{1, \ldots, J\}$ defines which object/topic is associated with patch i in image m, defined by $\boldsymbol{x}_{mi}$. Finally, the vector of probabilities $\boldsymbol{\pi}_{h_{mi}}$ defines the associated probabilities with which columns of $\mathbf{D}$ ("image words") are used.

4.2.3 Handling words/annotations

If annotations are available for at least a subset of the M images, it is desirable to leverage the information they provide. For each image class $t \in \{1, \ldots, T\}$ there is a unique distribution over the L words, and therefore the observed count of words for

image m (when words are available) is drawn

$$\begin{aligned} \boldsymbol{y}'_m &\sim \text{Mult}(\boldsymbol{\omega}_{r_m}, N_m) \\ \boldsymbol{\omega}_t &\sim \text{Dir}(\alpha_\omega/L, \cdots, \alpha_\omega/L) \end{aligned} \tag{4.4}$$

where $\boldsymbol{y}'_m = \boldsymbol{y}_m N_m/|\boldsymbol{y}_m|$ and $|\boldsymbol{y}_m|$ represents the total number of words associated with image m. Recall that r_m is the topic/class associated with image m.

Note that we have scaled the observed count of words $\boldsymbol{y}_m$ to produce $\boldsymbol{y}'_m$, and the total number of words used in $\boldsymbol{y}'_m$ equals N_m, the number of image patches used in the analysis of image m. This has been found important in our numerical studies, as it places the image features and words on equal footing, when words are present. Typically $|\boldsymbol{y}_m| \ll N_m$, and therefore if this rescaling is not performed the contribution to the likelihood from the image features far overwhelms the likelihood contribution from the words. This rescaling of the word count is equivalent to raising the multimonial contribution to the likelihood function from $\boldsymbol{y}_m$ by power $N_m/|\boldsymbol{y}_m|$.

A graphical representation of the model is summarized in Fig. 5.1, in which shaded and unshaded nodes indicate observed and latent variables, respectively. An arrow indicates dependence between variables. The boxes denote repetition, with the number of repetitions indicated by the variables in the corner of boxes.

4.2.4 Discussion

While the hierarchical form of the model may appear relatively complicated, we have found it to be robust and relatively insensitive to parameter settings. There has been no tuning performed for any hyperparameters to achieve the results presented below, with parameters set in a "standard" way for such models. Specifically, the hyperparameters for the gamma distributions on the precisions were set as $(10^{-6}, 10^{-6})$. For the hierarchical beta process we set $c_0 = 10$, $\eta_0 = 0.5$ and $c_1 = 1$. The parameters on the Dirichlet distributions were set as $\alpha_\nu = 1$, $\alpha_\mu = 1$ and $\alpha_\omega = 1$.

The manner in which annotations are handled in the proposed model is more flexible than how such were considered in Du et al. (2009). Specifically, in the latter paper a single word was associated with each object class in the scene, and therefore the number of objects J was required to be equal to the number of words L. In our model J and L are in general different, and the number of inferred objects need not be equal to the number of words; this implies that multiple words may be used to represent the same object type.

In the course of developing the proposed model, we considered different details on the model construction. For example, we considered a stick-breaking representation for the beta process, with Teh et al. (2007) $\eta_k = \prod_{i=1}^{k} u_l$ with $u_l \sim \text{Beta}(\beta, 1)$. The advantage of this construction is that it associates the important (large) η_k with small indices k. This is of interest particularly when truncating the beta process to K atoms, as done here. We found that the model above worked the same as when the stick-breaking form of the beta process was employed, and therefore the former was adopted for its simplicity.

Note that each image was above assumed associated with a particular class, with an image class defined by a distribution over topics, $\boldsymbol{\nu}_t$. This was done to address the specific applications discussed below, of image clustering. In this setting all images in class r_m share the same distribution over topics, $\boldsymbol{\nu}_{r_m}$. In typical topic models Blei et al. (2003b) each image has a unique distribution over topics, and this may also be considered here if desired. In this case rather than clustering images via the indicator r_m, each image may have a unique distribution over topics, drawn for example from a hierarchical Dirichlet process Teh et al. (2004a). We also considered drawing the probability vector $\boldsymbol{\mu}$ over image categories via a stick-breaking representation Sethuraman (1994b) rather than from a Dirichlet distribution, with results similar to those reported below.

4.3 Model Inference

Because all consecutive layers except for η_k in the hierarchical model are in the conjugate-exponential family, we employ Gibbs sampling for each parameter except η_k, for which slice sampling is utilized in Zhou et al. (2011c). When sampling $\boldsymbol{s}$, instead of drawing from a normal distribution as in (Zhou et al., 2009), we draw from the *truncated* normal distribution (we also considered the normal distribution, and the truncated version provided improved results). Below we briefly summarize update equations for unique aspects of the proposed model:

Sampling $\boldsymbol{d}_k$: The posterior of the kth dictionary atom $\boldsymbol{d}_k$ can be shown to be normal with covariance $\boldsymbol{\Sigma}_{\boldsymbol{d}_k}$ and mean $\boldsymbol{\mu}_{\boldsymbol{d}_k}$

$$\boldsymbol{\Sigma}_{\boldsymbol{d}_k} = (P + \gamma_\epsilon \sum_{m=1}^{M} \sum_{i=1}^{N_m} z_{mik}^2 s_{mik}^2)^{-1} \tag{4.5}$$

$$\boldsymbol{\mu}_{\boldsymbol{d}_k} = \gamma_\epsilon \boldsymbol{\Sigma}_{\boldsymbol{d}_k} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \widetilde{\boldsymbol{x}}_{mi}^{-k} s_{mik} s_{mik} \tag{4.6}$$

where $\widetilde{\boldsymbol{x}}_{mi}^{-k} = \boldsymbol{x}_{mi} - \mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}) + \boldsymbol{d}_k(z_{mik} \odot s_{mik})$.

Sampling z_{mik}: For the ith patch in the mth image, sample the binary sparse code $\boldsymbol{z}_{mi} = (z_{mi1}, ..., z_{miK})$ with

$$P(z_{mik} = 1) = \pi_{h_{mi}k} \exp[-\frac{\gamma_\epsilon}{2}(\boldsymbol{d}_k^T \boldsymbol{d}_k s_{mik}^2 - 2\boldsymbol{d}_k^T s_{mik} \widetilde{\boldsymbol{x}}_{mi}^{-k})] \tag{4.7}$$

$$P(z_{mik} = 0) = 1 - \pi_{h_{mi}k} \tag{4.8}$$

Thus whether the kth dictionary atom will be chosen for the ith patch of the mth image is drawn $z_{mik} \sim \text{Bernoulli}(\frac{P(z_{mik}=1)}{P(z_{mik}=1)+P(z_{mik}=0)})$.

Sampling s_{mik}: The posterior of positive weight s_{mik} can be obtained as a

truncated normal distribution, with covariance $\boldsymbol{\Sigma}_{s_{mik}}$ and mean $\boldsymbol{\mu}_{s_{mik}}$, where

$$\boldsymbol{\Sigma}_{s_{mik}} = 1/(\gamma_s + \gamma_\epsilon \boldsymbol{d}_k^T \boldsymbol{d}_k z_{mik}^2) \tag{4.9}$$

$$\boldsymbol{\mu}_{s_{mik}} = \gamma_\epsilon \boldsymbol{\Sigma}_{s_{mik}} \boldsymbol{d}_k^T z_{mik} \widetilde{\boldsymbol{x}}_{mi}^{-k} \tag{4.10}$$

Sampling $\boldsymbol{\pi}_j$: Given a Beta$(\boldsymbol{\pi}; c_1\boldsymbol{\eta}, c_1(1-\boldsymbol{\eta}))$ prior where $\boldsymbol{\eta} = [\eta_1, \eta_2, ..., \eta_K]^T$, the posterior of dictionary usage for the jth object is $p(\boldsymbol{\pi}_j|-) = \text{Beta}(\boldsymbol{\pi}_j; \psi_{1j}, \psi_{2j})$, where

$$\psi_{1j} = c_1\boldsymbol{\eta} + \sum_{m=1}^{M}\sum_{i=1}^{N_m} \delta(h_{mi} = j)\boldsymbol{z}_{mi} \tag{4.11}$$

$$\psi_{2j} = c_1(1-\boldsymbol{\eta}) + \sum_{m=1}^{M}\sum_{i=1}^{N_m} \delta(h_{mi} = j)(1-\boldsymbol{z}_{mi}) \tag{4.12}$$

dictionary atoms are globally shared; thus, for all patches, the summation of dictionary usage over those patches that belong to the jth topic contributes to the dictionary usage of the jth object.

Sampling h_{mi}: The conditional distribution of an object index for the ith patch of the mth image depends on the prior and its corresponding dictionary usage. h_{mi} is sampled from a J-dimensional multinomial distribution as:

$$P(h_{mi} = j|-) \propto \nu_{r_m j} \prod_{k=1}^{K} \pi_{jk}^{z_{mik}} (1-\pi_{jk})^{1-z_{mik}} \tag{4.13}$$

Sampling r_m : The conditional distribution of a category index for a particular image depends on 1) the prior, 2) the likelihood of topic distribution over all patches within the image, 3) the likelihood of annotations associated with the image. The category indicator r_m is sampled from a T-dimensional multinomial distribution as:

$$P(r_m = t|-) \propto \mu_t \prod_{j=1}^{J} \nu_{tj}^{\sum_{i=1}^{N_m}\delta(h_{mi}=j)} \prod_{l=1}^{L} \omega_{tl}^{y'_{ml}} \tag{4.14}$$

Sampling ν_{tj}, ω_{tl}, and μ_t : The prior has $p(\nu_{tj}|-) = \text{Dir}(\nu^*_{t1}, ..., \nu^*_{tJ})$, $p(\omega_{tl}|-) = \text{Dir}(\omega^*_{t1}, ..., \omega^*_{tL})$ and $p(\mu_t|-) = \text{Dir}(\mu^*_1, ..., \mu^*_T)$, where Because of conjugacy, we may infer their Dirichlet distributed posteriors, each element of which is expressed as

$$\nu^*_{tj} = \frac{\alpha_\nu}{L} + \sum_{m=1}^{M} \big[\sum_{i=1}^{N_m} \delta(h_{mi} = j)\big]\delta(r_m = t) \tag{4.15}$$

$$\omega^*_{tl} = \frac{\alpha_\omega}{L} + \sum_{m=1}^{M} \delta(r_m = t) y'_{ml} \tag{4.16}$$

$$\mu^*_t = \frac{\alpha_\mu}{T} + \sum_{m=1}^{M} \delta(r_m = t). \tag{4.17}$$

Sampling η_k: The posterior of η_k can be expressed as

$$p(\eta_k|-) \propto \text{Beta}(\eta_k; c_0\eta_0, c_0(1-\eta_0)) \prod_{j=1}^{J} \text{Beta}(\pi_{jk}; c_1\eta_k, c_1(1-\eta_k)) \tag{4.18}$$

Noting when $c_1 = 1$, with the Euler's reflection formula $\Gamma(1-x)\Gamma(x) = \pi/\sin(\pi x)$, this posterior then yields to

$$p(\eta_k|-) \propto \eta_k^{c_0\eta_0 - 1}(1-\eta_k)^{c_0(1-\eta_0)-1} \sin^J(\pi\eta_k) \exp(\eta_k \sum_{j=1}^{J} \log \frac{\pi_{jk}}{1-\pi_{jk}}) \tag{4.19}$$

With slice sampling, for the current tth iteration, we introduce three latent variables

$$\begin{aligned} \rho^{(t)}_{1k} &\sim \text{Unif}(0, \eta_k^{(t-1)} c_0\eta_0 - 1) \\ \rho^{(t)}_{2k} &\sim \text{Unif}(0, \sin^J(\pi\eta_k^{(t-1)})) \\ \rho^{(t)}_{3k} &\sim \text{Unif}(0, (1-\eta_k^{(t-1)})^{c_0(1-\eta_0)-1}). \end{aligned} \tag{4.20}$$

From (4.20), we may derive the range of $\eta_k^{(t)}$in terms of $\rho^{(t)}_{1k}$, $\rho^{(t)}_{2k}$ and $\rho^{(t)}_{3k}$. For brevity, this range is denoted as $\mathbf{I}(\eta_k^{(t)})$, with which we may draw $\eta_k^{(t)}$ from the truncated exponential distribution as

$$\eta_k^{(t)} \sim \exp(-\sum_{j=1}^{J} \log \frac{\pi_{jk}}{1-\pi_{jk}}) \mathbf{I}(\eta_k^{(t)}) \tag{4.21}$$

The above inference for η_k is based on the value $c_1 = 1$, which yields to efficient computation. However, other cases of c_1 can also be achieved by applying the random-walk Metropolis-Hastings.

4.4 Experimental Results

We test our model with one relatively simple but illustrative dataset (MNIST handwritten digits) and three real-world image data sets (MSRC, LabelMe and UIUC-Sport); the latter three contain annotations. For all experiments, we process patches from each image. For the MNIST data we randomly select 50 partially overlapping patches in each image, with 15×15 patch size, and for the other three datasets we collect all $32 \times 32 \times 3$ non-overlapping patches from the color image (we could also consider overlapping patches in this case, but it was found unnecessary). These patches are used to constitute the data matrix $\mathbf{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$, where $\boldsymbol{x}_i \in \mathbb{R}^P$, with P the number of pixels in each patch ($P = 225$ for MNIST, and $P = 3072$ for the other three data); N is the total number of patches in the dataset. The matrix $\mathbf{X}$ is pre-whitened with principal component analysis (PCA) and the first 200 principle components are employed (200 keeps about 95% of the energy of the original data, achieving a good balance between accuracy and complexity). To initialize the dictionary, we can use random initialization or some fixed redundant bases, such as over-completed DCT. In this paper, we use the covariate-dependent HBP (with the covariates linked to the relative locations between data samples) to learn an initial set of dictionary atoms, which are found to match the local latent features Zhou et al. (2011c).

In all experiments we set the truncation levels as $K = 400$, $J = 100$ and $T = 30$. Similar results were found for larger truncations. Note that these truncation levels are upper bounds on the associated parameter, while the model infers the number

of components needed. For each experiment, we run 1000 MCMC iterations, and collect the last 500 samples.

4.4.1 MNIST Handwritten Digits

For the MNIST handwritten digit database, we randomly choose 100 samples per digit (digits 0 through 9), and therefore 1000 samples are considered in total; the original digit images are of size 28×28. In this experiment annotations are not considered.

Each collection sample manifests a number of unique image classes, and often more than 10 classes are inferred, since some digits tend to occupy more than one image class (as a consequence of different styles of writing the digits). Fig. 4.2 displays five random examples associated with each image class inferred, at a typical collection sample. From Fig. 4.2 we see that there is more than one way some digits may be expressed, and the different writing styles constitute unique image classes inferred by the model.

As seen from Fig. 4.2, the inferred clusters are readily labeled in terms of truth, based upon the large frequency with which a particular cluster is associated with one digit. In Fig. 4.3(a) we present a confusion matrix, which quantifies the probability that a given digit is clustered "properly", in the sense that it is in a cluster dominated by the same digit type (this quantifies the "purity" of the clusters, in the context of being associated with the same image type). The average clustering accuracy is 81.4%, and we note that this performance is achieved with an *unsupervised* model, with dictionary learning and clustering performed simultaneously.

4.4.2 Microsoft Data

The experiments with the MNIST data demonstrate the ability of the model to cluster images accurately; henceforth we do such in the presence of annotations,

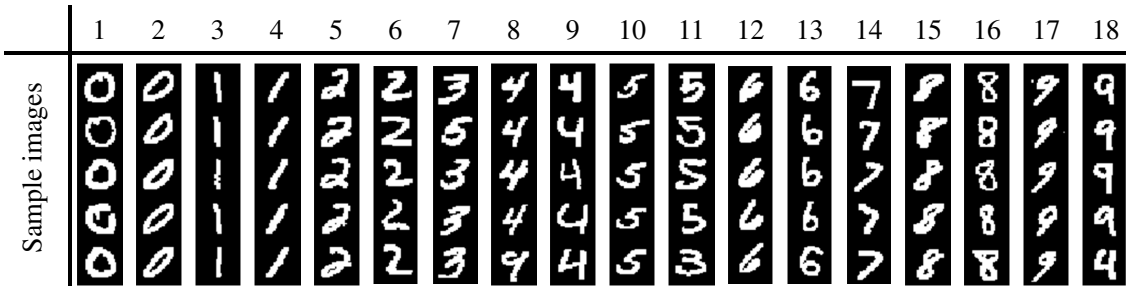
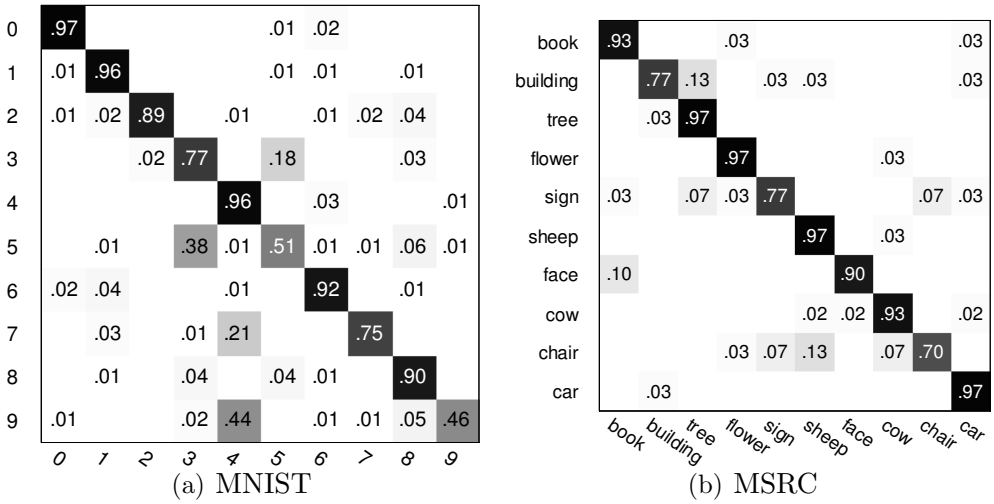


FIGURE 4.2: Example images associated with 18 inferred classes, with each column representing one unique class.



(a) MNIST (b) MSRC

FIGURE 4.3: (a) Confusion matrix of MNIST data, with average accuracy of 81.04%. (b) Confusion matrix of MSRC data, with average accuracy of 89.06%.

considering natural images. We use the same settings of images and annotations from the MSRC data[1] as considered in Du et al. (2009), to allow a direct comparison. We choose 320 images from 10 categories of images with manual annotations available. The categories are "tree", "building", "cow", "face", "car", "sheep", "flower", "sign", "book" and "chair". The numbers of images are 45 and 35 in the "cow" and "sheep" classes, respectively and 30 in all the other classes. Each image has size 213×320 or 320×213. For annotations, we remove all annotation-words that occur less than 8 times (approximately 1% of them), and obtain 15 unique annotation-words, thus $L =$

[1] *http://research.microsoft.com/en-us/projects/objectclassrecognition/*

15. For each category, we randomly choose 10 images, and remove their annotations, treating them as non-annotated images within the analysis.

We inferred 11 clusters, and found that the "chair" image class is divided into two types. Using such labeling of clusters based on truth, we may constitute a confusion matrix, defining the probability that an image from a given class is associated with the appropriate mixture component, as was done with the MNIST data. The confusion matrix as computed form the collection samples is depicted in Fig. 4.3(b). The average accuracy is 89.06%, outperforming the results in Du et al. (2009) by 6.16% under the same test settings (note that in Du et al. (2009) predefined features were extracted from super-pixels, and VQ was employed). By contrast, the proposed model does image clustering (topic modeling) and feature design simultaneously, without VQ. Fig. 4.4 shows three example images correctly assigned to each of the clusters. In Fig. 4.4 we observe that many of the "inaccurate" classifications that cause errors in Fig. 4.3(b) actually make a lot of sense. For example the "face" image at the top-right in Fig. 4.4 is "incorrectly" assigned to the "book" class, as a consequence of the books in the background of the face picture. As another example, sheep are misclassified as cows.

Each image class is characterized by a distribution over objects, and these objects may be linked to words via the annotation, when available. A good connection is inferred between words and image classes (clusters). Based on the learned posterior word distribution $\boldsymbol{\omega}_t$ for the tth image class, we can further infer which words are most probable for each image class (category). Figure 4.5 shows the $\boldsymbol{\omega}_t$ for 9 classes, with the five largest-probability words displayed.

FIGURE 4.4: Example images inferred for each class. Each row is for one category. The first three columns on the left show 3 examples of correctly inferred images, the last column on the right shows an example of incorrectly recognized image.

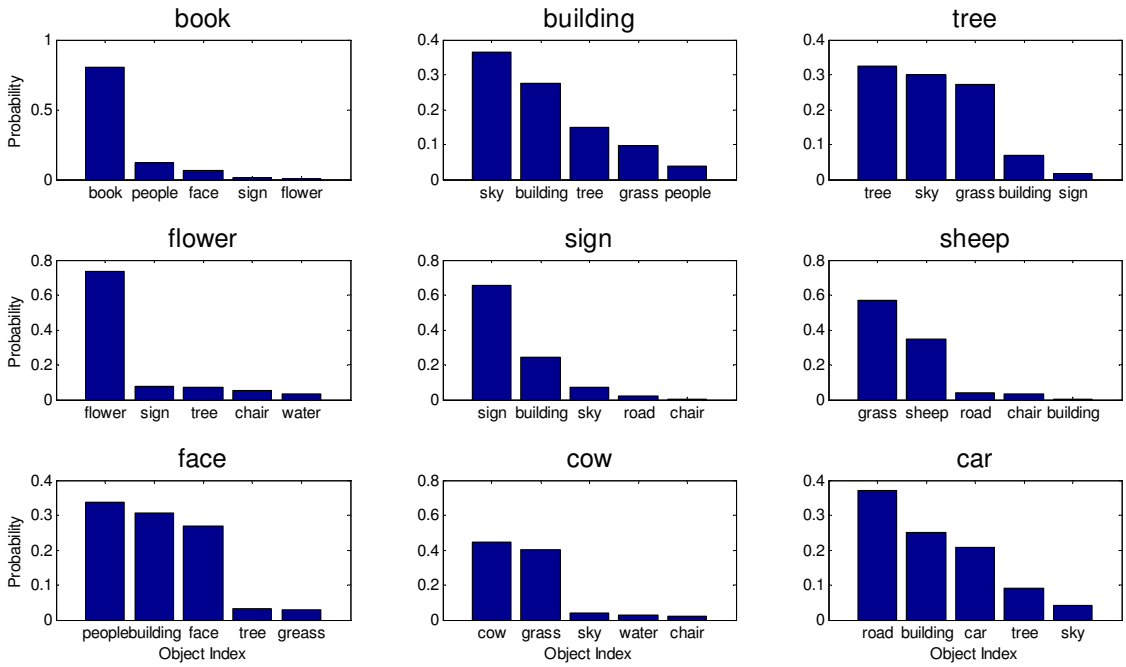


FIGURE 4.5: Each image class is characterized by a distribution over objects, and these objects may be linked to words via the annotation, when available. For the MSRC data we display the word probabilities for inferred image types. We may therefore connect words to the classes, with the first row reflecting "book", "building" and "tree" categories, for example.

, with no further details here, for brevity. Below we show detailed word associations for the UIUC-Sport data.

For the above results, the dictionary is performed simultaneously with topic modeling, with the dictionary learning performed directly on image patches (below we refer to this as "online"). As comparisons, we consider the following alternatives. In one test, the dictionary atoms, initialized with the method discussed in Zhou et al. (2011c), are fixed, and we use the dictionary in the topic model as before (below we refer to this as "offline"). This permits us to examine the benefit of simultaneously doing dictionary learning and topic modeling, which allows the dictionary atoms to be matched to the topic-modeling objective. As another example, topic modeling and dictionary learning are performed simultaneously, but the dictionary learning is

performed using SIFT features extracted from the same local region patches used in the previous dictionary learning. Finally, we remove dictionary learning altogether, and learn a codebook of dimension $K = 400$ (consistent with the dictionary-learning truncation level), with VQ codebook design performed directly on the image patches. The quantitative comparisons between these tests are summarized in Table 4.1. It is observed that dictionary learning performed directly on the patches yields best results, with an improvement manifested by the full online analysis (joint topic modeling and dictionary learning). There is a marked improvement in doing dictionary learning directly on the image patches, compared to doing such on the SIFT features.

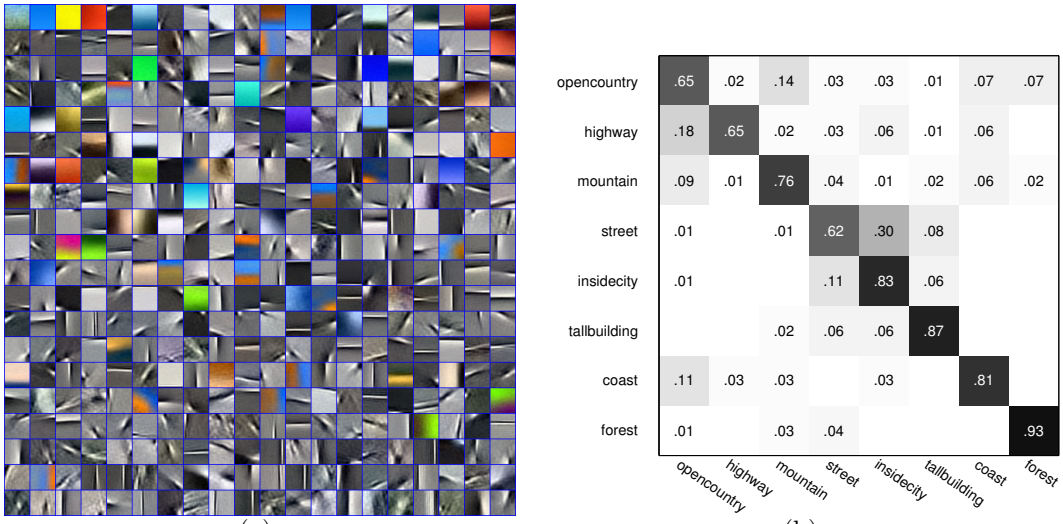
Table 4.1: Performance comparisons with different settings of features and dictionary, for the MSRC data.

Feature	Dictionary setting	Accuracy
Image patches	Online learning	89.06%
Image patches	Offline learning	87.50%
Image patches	K-means	67.81%
SIFT	Online learning	80.94%

4.4.3 LabelMe Data

We next consider the LabelMe dataset together with annotations[2]. The LabelMe data contain 8 image classes: "coast", "forest", "highway", "inside city", "mountain", "open country", "street" and "tall building". We use the same settings of images and annotations as Wang et al. (2009): we randomly select 200 images for each class, thus the total number of images is 1600. Each image is resized to be 256×256 pixels. For the annotations, we remove terms that occur less than 3 times, and obtain a vocabulary of 186 unique words, thus $L = 186$. There are 6 terms per annotation in the LabelMe data on average. We then randomly select 800 images, and remove their annotations treating them as non-annotated images, so that the total set of images analyzed are partially annotated, as for the MSRC example.

[2] *http://www.cs.princeton.edu/ chongw/*



(a) (b)

FIGURE 4.6: Results for the LabelMe data. (a) The inferred dictionary with elements sorted in a decreasing order, (b) confusion Matrix over the 800 non-annotated images, with the average performance of 76.25%.

Fig. 4.6(a) shows the inferred dictionary atoms, demonstrating both color and texture features. The model inferred 13 image classes, and 77 unique objects/topics. Although the model is learned using both annotated and non-annotated images, we focus on the confusion matrix for the 800 non-annotated images in Fig. 4.6(b), computed as above (each of the inferred image classes may be unambiguously associated with one of the true classes). In Table 4.2 we summarize average clustering accuracy on the annotated and non-annotated images, with results also summarized there for the UIUC-Sport data we consider next. In Table 4.2 we also provide a comparison to results from Wang et al. (2009).

Table 4.2: Performance comparisons of confusion matrix. 'annotated' and 'non-annotated' separately denote the accuracy of confusion matrix computed over the annotated images and the non-annotated images. 'Wang' represents the result reported in Wang et al. (2009).

	annotated	non-annotated	Wang
LabelMe	92.25%	76.25%	76%
UIUC-Sport	91.03%	69.11%	66%

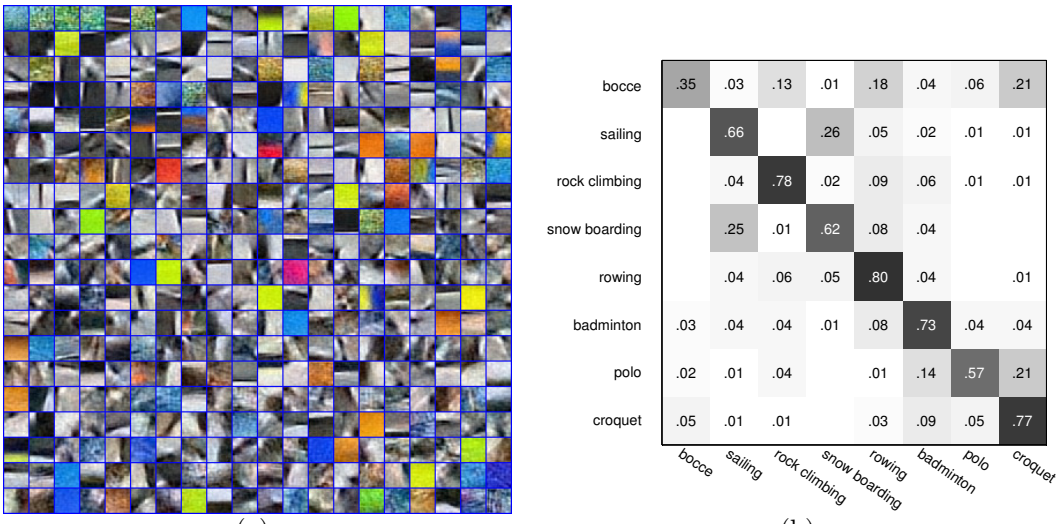
4.4.4 UIUC-Sport Data

Finally we test our model on the UIUC-Sport dataset. The UIUC-Sport dataset contains 8 types of sports: "badminton"(200 images), "bocce"(137 images), "croquet"(236 images), "polo"(182 images), "rock climbing"(194 images), "rowing"(250 images), "sailing"(190 images), and "snow boarding"(190 images). The total number of images is 1579. With the purpose of comparison, we use the same settings of images as Wang et al. (2009)[3]. Since the tags contain too many arbitrarily noisy words, we first obtain candidate tags belonging to 'physical entity' Li et al. (2009) by using WordNet synsets[4], and then select the 30 most frequent words from these candidate tags; thus $L = 30$. We evenly split each class and remove annotations of half, treating them as non-annotated images. The inferred dictionary and confusion matrix is shown in Fig. 4.7, with the average accuracy of 69.11%, summarized in Table 4.2. Based on the learned posterior word distribution $\boldsymbol{\omega}_t$ for the tth image class, we can further infer which words are most probable for each image class (category). Fig. 4.8 shows the $\boldsymbol{\omega}_t$ for 8 classes, with the five largest-probability words displayed. A good connection is manifested between the words and image classes. The model clearly learns a good statistical distribution over words, matched to the latent image class/category. Further, the confusion matrices demonstrate that the model can infer the image class well. Therefore, the model performs well in *statistically* annotating non-annotated images (not further detailed, for brevity). The presence of the annotations assists with the clustering of the images into categories. Linkages are inferred between objects in the images and associated words (when present), and this assists clustering of images, even for those images without annotations.

The experiments above have been performed in 64-bit Matlab on a machine with

[3] The total number reported in the paper is 1792. According to the resources that also provided in the paper (*http://vision.stanford.edu/lijiali/*), there are actually 1579 images available.

[4] *http://wordnet.princeton.edu/*



(a) (b)

FIGURE 4.7: For the UIUC-Sport data, (a): The inferred dictionary with elements sorted in a decreasing order of importance. (b): Confusion Matrix over the 688 non-annotated images.

2.27 GHz CPU and 4Gbyte RAM. One MCMC run of the proposed model takes around 5, 2, 11 and 10 minutes respectively for the MNIST, MSRC, LabelMe and UIUC experiments (in which we simultaneously analyzed respectively 1000, 320, 1600, and 1579 total images). The proposed model could also be implemented via variational Bayesian (VB) analysis, that may yield to efficiency.

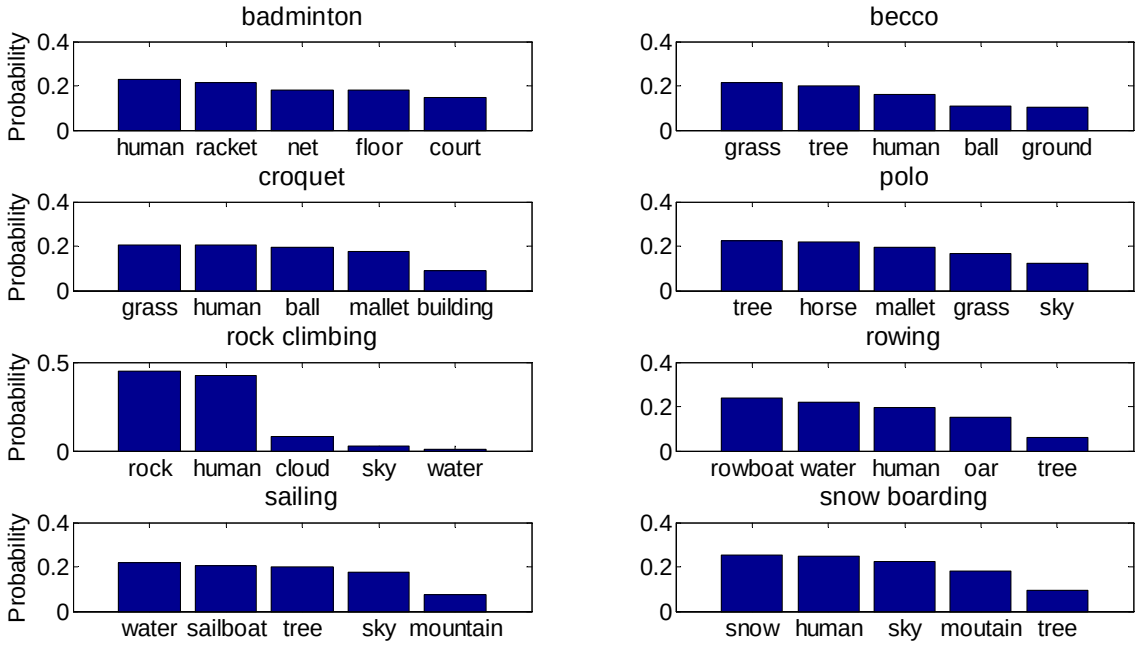


FIGURE 4.8: Inferred distributions over words for UIUC-Sport data, as a function of inferred image category. Names on the horizontal represents the annotation terms, the order of which varies across the categories. The vertical axis represents the distribution.

4.5 Summary

A new model has been developed to integrate topic modeling and dictionary learning into a unified Bayesian setting. In comparison with previous models, based on image features which were carefully defined (e.g., superpixels, SIFT, shape, texture, etc.), the proposed model achieves performance as good or better as existing published results. This is realized by executing the dictionary-learning component of the model directly on patches from the original image. The model is therefore not specialized to imagery, and may be applied to other problems, for example annotated audio signals.

5

Nested Dictionary Learning for Hierarchical Organization of Imagery and Text

5.1 Introduction

In Chapter 4, a new nonparametric Bayesian model is developed to integrate dictionary learning and topic model into a unified framework. A dictionary-learning framework is employed to eliminate the need to perform VQ. This dictionary learning could be applied to traditional features pre-computed from the image, or it could be applied directly to patches of raw imagery, thereby ameliorating the requirement of separating the feature-design and topic-modeling steps. The model is employed to analyze partially annotated images, with the dictionary learning performed directly on image patches. However, like most of the image-based topic modeling (Du et al., 2009; Li et al., 2009; Wang et al., 2009), it is performed at a single scale or level, thereby not accounting for the hierarchical characteristics of most natural imagery.

Recently Li et al. (2010) employed a nested Chinese restaurant process (nCRP) to infer a hierarchical tree representation for a corpus of images and (if available) accompanying text; however, in that work the VQ step was still employed, and

therefore a precomputation of features was as well. Further, in Li et al. (2010), while the tree width was inferred, the depth was set. Finally, the nCRP construction in Li et al. (2010) has the disadvantage of only updating parent-child-transition parameters from one node of the tree at a time, in a sampler, yielding poor mixing relative to a stick-breaking Dirichlet process (DP) implementation (Ishwaran and James, 2001).

Motivated by these recent contributions, and the limitations of most existing topic models of imagery and text discussed in Chapter 4, this chapter makes the following contributions:

- A nested DP (nDP) model is developed to learn a hierarchical tree structure for a corpus of imagery and text, with a stick-breaking construction employed; we infer both the tree depth and width, using a retrospective stick-breaking construction (Papaspiliopoulos and Roberts, 2008).

- A beta-Bernoulli dictionary learning framework (Zhou et al., 2011c) is adapted to such a hierarchical model, removing the VQ step, and allowing one to perform topic modeling directly on image patches, thereby integrating feature design and topic modeling. However, if desired, the dictionary learning may also be applied to features pre-computed from the image, using *any* existing method for feature design, and again removing the limitations of VQ.

To summarize up, a tree-based dictionary learning model is developed for joint analysis of imagery and associated text. The dictionary learning may be applied directly to the imagery from patches, or to general feature vectors extracted from patches or superpixels (using any existing method for image feature extraction). Each image is associated with a path through the tree (from root to a leaf), and each of the multiple patches in a given image is associated with one node in that path. Nodes near the tree root are shared between multiple paths, representing image

characteristics that are common among different types of images. Moving toward the leaves, nodes become specialized, representing details in image classes. If available, words (text) are also jointly modeled, with a path-dependent probability over words. The tree structure is inferred via a nested Dirichlet process, and a retrospective stick-breaking sampler is used to infer the tree depth and width.

5.2 Modeling Image Patches

We wish to build a hierarchical model to arrange M images and their associated annotations (when available); the vocabulary of such annotations is assumed to be of dimension N_v. The vector $\boldsymbol{x}_{mi}$ represents the pixels or features associated with the ith patch in image m, and $\boldsymbol{y}_m = (y_{m1}, \ldots, y_{mN_v})^T$ represents a vector of word counts associated with that image, when available (y_{mn} represents the number of times word $n \in \{1, \ldots, N_v\}$ is present in the annotation).

The mth image is divided into N_m patches (or superpixels (Li et al., 2010)), and the data for the ith patch is denoted $\boldsymbol{x}_{mi} \in \mathbb{R}^P$ with $i = 1, \ldots, N_m$. The vector $\boldsymbol{x}_{mi}$ may represent raw pixel values, or a feature vector extracted from the pixels (using any available method of image feature extraction, *e.g.*, SIFT (Lowe, 1999)).

Each $\boldsymbol{x}_{mi}$ is represented as a sparse linear combination of learned dictionary atoms. Further, each patch is assumed associated with a "topic"; the probability of which dictionary atoms are employed for a given patch is dictated by the topic it is associated with.

Specifically, each patch is represented as $\boldsymbol{x}_{mi} = \mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}) + \boldsymbol{e}_{mi}$, where $\odot$ represents the element-wise/Hadamard product, $\mathbf{D} = [\boldsymbol{d}_1, \cdots, \boldsymbol{d}_K] \in \mathbb{R}^{P \times K}$, K is the truncation level on the possible number of dictionary atoms, $\boldsymbol{z}_{mi} = [z_{mi1}, \cdots, z_{miK}]^T$, $\boldsymbol{s}_{mi} = [s_{mi1}, \cdots, s_{miK}]^T$, $z_{mik} \in \{0, 1\}$ indicates whether the kth atom is *active* within patch i in image m, $s_{mik} \in \mathbb{R}^+$, and $\boldsymbol{e}_{mi}$ is the residual. Note that $\boldsymbol{z}_{mi}$ represents the specific sparseness pattern of dictionary usage for $\boldsymbol{x}_{mi}$. The hierarchical form of the

model is

$$\begin{aligned} \boldsymbol{x}_{mi} &\sim \mathcal{N}(\mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}), \gamma_e^{-1}\mathbf{I}_P) \\ \boldsymbol{d}_k &\sim \mathcal{N}(0, \frac{1}{P}\mathbf{I}_P) \\ \boldsymbol{s}_{mi} &\sim \mathcal{N}_+(0, \gamma_s^{-1}\mathbf{I}_K) \\ \boldsymbol{z}_{mi} &\sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{h_{mi}k}) \end{aligned} \tag{5.1}$$

where gamma priors are placed on both γ_e and γ_s. Positive weights $\boldsymbol{s}_{mi}$ (truncated normal, $\mathcal{N}_+(\cdot)$) are imposed, which we have found to yield improved results.

The indicator variable h_{mi} defines the topic associated with $\boldsymbol{x}_{mi}$. The K-dimensional vector $\boldsymbol{\pi}_h$ defines the probability that each of the K columns of $\mathbf{D}$ is employed to represent topic h, where the kth component of $\boldsymbol{\pi}_h$ is π_{hk}. These probability vectors are drawn

$$\boldsymbol{\pi}_h \sim G_0, \quad G_0 = \prod_{k=1}^{K} \text{Beta}(a_0/K, b_0(K-1)/K) \tag{5.2}$$

where π_{hk} represents the probability of using $\boldsymbol{d}_k$ for object type h, and the introduction of G_0 is for discussions below. This representation for $\boldsymbol{\pi}_h$ corresponds to an approximation to the beta-Bernoulli process (Paisley and Carin, 2009c; Thibaux and Jordan, 2007a; Zhou et al., 2011c), which also yields an approximation to the Indian buffet process (IBP) (Griffiths and Ghahramani, 2005a; Teh et al., 2007).

5.3 Tree Structure via nDP

The nested Dirichlet process (nDP) tree construction developed below is an alternative means of constituting the same type of tree manifested by the nested Chinese restaurant process (Blei et al., 2003a; Li et al., 2010). We emphasize the nDP construction because of the stick-breaking implementation we employ, which allows block

updates, and therefore often manifests better mixing than the nCRP-type implementation (Ishwaran and James, 2001). Related work was considered in Wang and Blei (2009), but VB inference was employed and the tree size was therefore not inferred (a fixed truncation was imposed). The retrospective sampler developed below allows inference of both the tree depth and width (Papaspiliopoulos and Roberts, 2008).

Consider a draw from a DP, $G \sim \text{DP}(\gamma, G_0)$, where $\gamma > 0$ is an "innovation" parameter, with G_0 defined in (5.2). Then the DP draw (Ishwaran and James, 2001) may be expressed as $G = \sum_{n=1}^{\infty} \lambda_n \delta_{\phi_n}$, where $\lambda_n = \nu_n \prod_{l<n}(1-\nu_l)$, $\nu_l \sim \text{Beta}(1, \gamma)$, and $\phi_n \sim G_0$; each ϕ_n corresponds to a topic, as in (5.2). Letting $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots)^T$, we denote the draw of $\boldsymbol{\lambda}$ as $\boldsymbol{\lambda} \sim \text{Stick}(\gamma)$.

5.3.1 Tree width

Using notation from Adams et al. (2010), let $\boldsymbol{\epsilon}$ represent a path through the tree, characterized by a sequence of parent-child nodes, and let $|\boldsymbol{\epsilon}|$ be the length of this path (total number of layers traversed). In addition to representing a path through the tree, $\boldsymbol{\epsilon}$ identifies a node at layer $|\boldsymbol{\epsilon}|$, *i.e.*, the node at the end of path $\boldsymbol{\epsilon}$. For node $\boldsymbol{\epsilon}$, let $\boldsymbol{\epsilon}\epsilon_i$, $i = 1, 2, \dots$, denote the *children* of $\boldsymbol{\epsilon}$, at level $|\boldsymbol{\epsilon}| + 1$. To constitute a distribution over the children nodes, we draw $G_{\boldsymbol{\epsilon}} \sim \text{DP}(\gamma, G_0)$, yielding $G_{\boldsymbol{\epsilon}} = \sum_{i=1}^{\infty} \lambda_{\boldsymbol{\epsilon}\epsilon_i} \delta_{\phi_{\boldsymbol{\epsilon}\epsilon_i}}$, where $\lambda_{\boldsymbol{\epsilon}\epsilon_i} = \nu_{\boldsymbol{\epsilon}\epsilon_i} \prod_{j=1}^{i-1}(1-\nu_{\boldsymbol{\epsilon}\epsilon_j})$, $\nu_{\boldsymbol{\epsilon}\epsilon_j} \sim \text{Beta}(1, \gamma)$, and $\phi_{\boldsymbol{\epsilon}\epsilon_i} \sim G_0$, with G_0 defined in (5.2); $\boldsymbol{\lambda}_{\boldsymbol{\epsilon}} = (\lambda_{\boldsymbol{\epsilon}\epsilon_1}, \lambda_{\boldsymbol{\epsilon}\epsilon_2}, \dots)^T$ is denoted as drawn $\boldsymbol{\lambda}_{\boldsymbol{\epsilon}} \sim \text{Stick}(\gamma)$. The probability measure $G_{\boldsymbol{\epsilon}}$ constitutes in principle an infinite set of children nodes, with $\lambda_{\boldsymbol{\epsilon}\epsilon_i}$ defining the probability of transiting from node $\boldsymbol{\epsilon}$ to child ϵ_i; $\phi_{\boldsymbol{\epsilon}\epsilon_i}$ constitutes the topic-dependent probability of dictionary usage at that child node.

The process continues *in principle* to an infinite number of levels, with each child node spawning an infinite set of subsequent children nodes, manifesting a tree of infinite depth and width. However, note that a draw $\boldsymbol{\lambda}_{\boldsymbol{\epsilon}}$ will typically only have a relatively small number of components with appreciable amplitude. This means that

while $G_{\boldsymbol{\epsilon}}$ constitutes in principle an infinite number of children nodes, only a small fraction will be visited with appreciable probability.

Let $\boldsymbol{c}_m = (c_m^1, c_m^2, \dots)^T$ represent the path associated with image m, where c_m^l corresponds to the node selected at level l. For conciseness we write $\boldsymbol{c}_m \sim \text{nCRP}(\gamma)$ (Blei et al., 2003a), emphasizing that the underlying transition probabilities $\boldsymbol{\lambda}_{\boldsymbol{\epsilon}}$, controlling the probabilistic path through the tree, are a function of parameter γ.

5.3.2 Tree depth

We also draw an associated probability vector $\boldsymbol{\theta}_m \sim \text{Stick}(\alpha)$. Patch $\boldsymbol{x}_{mi}$ is associated with level l_{mi} in path $\boldsymbol{c}_m$, where $l_{mi} \sim \sum_{l=1}^{\infty} \theta_{ml}\delta_l$. Since $\boldsymbol{\theta}_m$ typically only has a small number of components with appreciable amplitude, the tree depth is also constrained.

5.3.3 Modeling words

In Section 5.2 we developed topic (node) dependent probabilities of atom usage; we now extend this to words (annotations), when available. A distribution over words may be associated with each topic (tree node) h. For topic h we may draw (Blei et al., 2003b)

$$\boldsymbol{\psi}_h \sim \text{Dir}(\frac{\eta}{N_v}, \dots, \frac{\eta}{N_v}) \tag{5.3}$$

where $\boldsymbol{\psi}_h$ is the distribution over words for topic h.

Recall that each image/annotation is associated with a path $\boldsymbol{c}_m$ through a tree, and $\boldsymbol{\theta}_m$ controls the probability of employing each node (topic) on that path. Let θ_{mh} represent the probability that node h is utilized, with $h \in \boldsymbol{c}_m$. Then the "collapsed" probability of word usage on this path, marginalizing out the probability of node selection, may be expressed as

$$\boldsymbol{\psi}_{\boldsymbol{c}_m} = \sum_{h \in \boldsymbol{c}_m} \theta_{mh}\boldsymbol{\psi}_h \tag{5.4}$$

A probability over words $\boldsymbol{\psi}_{\boldsymbol{c}_m}$ is therefore associated with each path $\boldsymbol{c}_m$. One may argue that the $\boldsymbol{\theta}_m$ used to control node usage for image patches should be different from that used to represent words; this is irrelevant in the final model, as a path-dependent $\boldsymbol{\psi}_{\boldsymbol{c}_m}$ is drawn directly from a Dirichlet distribution (discussed below), and therefore (5.4) is only illustrative/motivating.

5.3.4 Retrospective sampling

The above discussion indicated that while the width and depth of the tree is infinite in principle, a finite tree is manifested given finite data, which motivates adaptive inference of the tree size. In a retrospective implementation of a stick-breaking process, we constitute a *truncated* stick-breaking process, denoted $\boldsymbol{w} \sim \text{Stick}_L(\gamma)$, with $w_n = V_n \prod_{l<n}(1 - V_l)$, $V_n \sim \text{Beta}(1, \gamma)$ for $n < L$, and $V_L = 1$; here there is an L-stick truncation, yielding $\boldsymbol{w} = (w_1, \ldots, w_L)^T$, with w_L representing the probability of selecting a stick *other* than sticks 1 through $L - 1$.

In a retrospective sampler (Papaspiliopoulos and Roberts, 2008), each of the aforementioned sticks is truncated as above. When drawing children nodes and levels, if the last stick (the Lth above) is selected, this implies that a new child/level must be added, since the first $L-1$ sticks are not enough to capture how the data are clustered. If stick L is selected, then a new node/level is constituted (a new child is added), by drawing a new $V_L \sim \text{Beta}(1, \gamma)$, and then $V_{L+1} = 1$, thereby now constituting an $(L+1)$-dimensional stick representation; the associated node-dependent statistics are constituted as discussed in Section 5.2 (drawing new probabilities over dictionary elements). The model therefore infers "in retrospect" that the L-level truncation was too small, and expands adaptively. The model also has the ability to shrink the number of sticks used at any component of the model, if less than the associated truncated level is needed to define the number of children/levels are actually utilized.

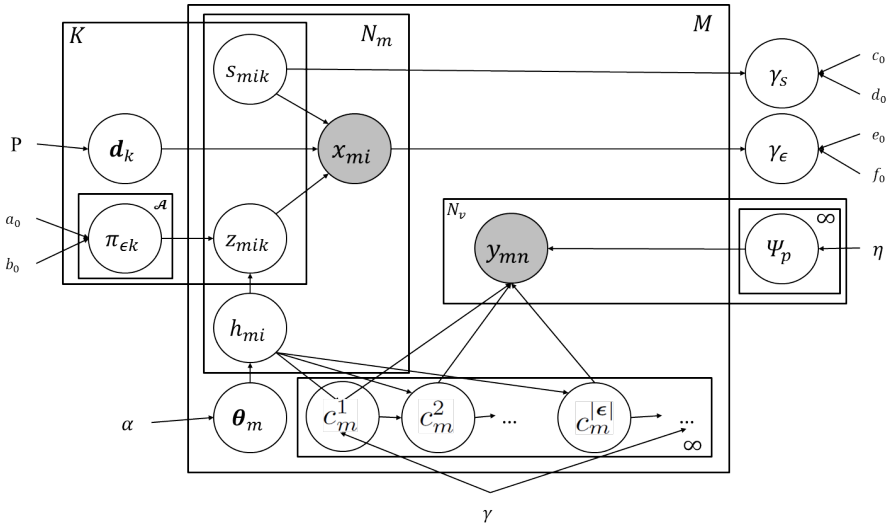


FIGURE 5.1: The graphical representation of the model.

5.3.5 Generative Process

Summarizing above, there is a stick-breaking representation for children nodes under each parent node, and for each node $\boldsymbol{\epsilon}$ there is a vector $\boldsymbol{\pi}_{\boldsymbol{\epsilon}}$ controlling the probability of using dictionary atoms. Each image m is associated with a path of the tree denoted as $\boldsymbol{c}_m$, and a stick-breaking draw $\boldsymbol{\theta}_m$ controlling to which depth patches are associated. Each patch i in image m can be assigned to different nodes of the path with the corresponding depth assignment l_{mi} determined by $\boldsymbol{\theta}_m$. We also associate a distribution over words for each path, when words (*e.g.*, annotations) are available. Figure 5.1 shows the graphical model. The generative process for the model is summarized as follows:

1. Draw dictionary $\mathbf{D} \sim \prod_{k=1}^{K} \mathcal{N}(0, \frac{1}{P}\mathbf{I}_P)$
2. Draw γ, α, γ_e and γ_s from respective gamma distributions
3. For each image $m \in \{1, 2, ..., M\}$
   (a) Draw $\boldsymbol{c}_m \sim \text{nCRP}(\gamma)$

(b) For each *newly utilized* node $\boldsymbol{\epsilon}$ in the tree, draw dictionary usage probabilities $\boldsymbol{\pi}_{\boldsymbol{\epsilon}} \sim \prod_{k=1}^{K} \text{Beta}(a_0/K, b_0(K-1)/K)$

(c) Draw $\boldsymbol{\theta}_m \sim \text{Stick}(\alpha)$

(d) For the ith patch or feature vector

i. Draw level index $l_{mi} \sim \sum_{l=1}^{\infty} \theta_{ml}\delta_l$, which along with $\boldsymbol{c}_{mi}$ defines node h_{mi}

ii. Draw $\boldsymbol{z}_{mi} \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{h_{mi}k})$, and $\boldsymbol{s}_{mi} \sim \mathcal{N}_+(0, \gamma_s^{-1}\mathbf{I}_K)$

iii. Draw $\boldsymbol{x}_{mi} \sim \mathcal{N}(\mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}), \gamma_e^{-1}\mathbf{I}_P)$

4. For each unique tree path p, draw $\boldsymbol{\psi}_p \sim \text{Dir}(\frac{\beta}{N_V}, \ldots, \frac{\beta}{N_V})$

5. If annotations are available for image m, $\boldsymbol{y}_m \sim \text{Mult}(|\boldsymbol{y}_m|, \boldsymbol{\psi}_{\boldsymbol{c}_m})$, where $|\boldsymbol{y}_m|$ is the total number of words in $\boldsymbol{y}_m$

In Step 3(b), new nodes (topics) are added "in retrospect", as discussed in the previous subsection (nodes may also be pruned with this sampler). After completing Step 3, the tree size is constituted, which allows Step 4, imposition of a distribution over words for each path. We may also place a distribution over words on each node/topic, as in (5.3), but for simplicity we here "collapse" this to a path-dependent distribution over words, as in (5.4).

5.4 Model Inference

A contribution of this paper concerns use of retrospective sampling to infer the tree width and depth. To save space for an extensive set of experimental results, we here only discuss updates associated with inferring the tree depth and width. A complete set of update equations are provided in Supplementary Material, where one may also find a summary of all notations in Table B.1.

5.4.1 Sampling level allocations

To sample l_{mi} from the conditional posterior, we first need to specify the likelihood that $\{\boldsymbol{\epsilon} \in \boldsymbol{c}_m\}$:

$$p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_{\boldsymbol{\epsilon}}, \boldsymbol{c}_m) = \prod_{k=1}^{K} \pi_{\boldsymbol{\epsilon}k}^{z_{mik}} (1 - \pi_{\boldsymbol{\epsilon}k})^{1-z_{mik}}$$

and the prior distribution, which is specified by a stick-breaking draw $\boldsymbol{\theta}_m$ for each image m. Although l_{mi} can be sampled from a closed form posterior for a fixed L_m, here to learn L_m adaptively we instead use an Metropolis-Hastings step, where the proposal distribution is defined as

$$q(l_{mi} = j) \propto \begin{cases} \theta_{mj} p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_j, \boldsymbol{c}_m), & j \leq L_m \\ \theta_{mj} \mathcal{M}_{mi}(L_m), & j > L_m \end{cases}$$

where $\mathcal{M}_{mi}(L_m) = \max_{1 \leq |\boldsymbol{\epsilon}| \leq L_m} \{p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_{\boldsymbol{\epsilon}}, \boldsymbol{c}_m)\}$. Note that the sampled value of l_{mi} is allowed to be larger than the truncation level L_m, consequently L_m and the depth of the tree is learned adaptively. The acceptance probability $\kappa_{mi}(j)$ for $l_{mi} = j$ is

$$\begin{cases} 1, & j \leq L_m \ \& \ L'_m = L_m \\ \min\{1, \frac{\tilde{c}_{mi}(L_m)\mathcal{M}_{mi}(L'_m)}{\tilde{c}_{mi}(L'_m)p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_{h_{mi}}, \boldsymbol{c}_m)}\}, & j \leq L_m \ \& \ L'_m < L_m \\ \min\{1, \frac{\tilde{c}_{mi}(L_m)p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_j, \boldsymbol{c}_m)}{\tilde{c}_{mi}(L'_m)\mathcal{M}_{mi}(L_m)}\}, & j > L_m \end{cases}$$

where the normalizing constant is defined as $\tilde{c}_{mi}(L_m) = \sum_{|\boldsymbol{\epsilon}|=1}^{L_m} \theta_{m|\boldsymbol{\epsilon}|} p(\boldsymbol{z}_{mi}|\boldsymbol{\pi}_{\boldsymbol{\epsilon}}, \boldsymbol{c}_m) + \mathcal{M}_{mi}(L_m)(1 - \sum_{|\boldsymbol{\epsilon}|=1}^{L_m} \theta_{m|\boldsymbol{\epsilon}|})$. The retrospective sampling procedure for l_{mi} is summarized in Algorithm 1.

5.4.2 Sampling paths

Given the level allocation samples $\{l_{mi}\}$, we sample the path assignment $\boldsymbol{c}_m$ for each image m. This consists of a sequence of parent-child transition sampling procedures: for a given parent node we wish to recursively sample a child node that it transitions to. We again adopt a retrospective sampling algorithm.

Algorithm 4 Retrospective Sampling for l_{mi}

Input: L_m, $\boldsymbol{z}$, $\boldsymbol{l}$, a_0, b_0
Output: l_{mi}, L_m
for $m = 1$ **to** M and $i = 1$ **to** N_m **do**
  Sample $\mu_{m|\boldsymbol{\epsilon}|}$, $\boldsymbol{\pi}_{\boldsymbol{\epsilon}}$ from the conditional posterior for $|\boldsymbol{\epsilon}| \leq L_m$, and from the prior for $|\boldsymbol{\epsilon}| > L_m$; $\theta_{m|\boldsymbol{\epsilon}|} = \mu_{m|\boldsymbol{\epsilon}|} \prod_{s=1}^{|\boldsymbol{\epsilon}|-1}(1-\mu_{ms})$
  Sample $U_{mi} \sim \text{Uniform}[0,1]$
  if $\sum_{s=1}^{j-1} q(l_{mi}=s) < U_{mi} \leq \sum_{s=1}^{j} q(l_{mi}=s)$ **then**
    Set $l_{mi} = j$ with probability $\kappa_{mi}(j)$, otherwise, leave l_{mi} unchanged
  else
    $L_m = L_m + 1$, set $l_{mi} = L_m$ with probability $\kappa_{mi}(L_m)$, otherwise, leave l_{mi} unchanged
  end if
end for

For a given node $\boldsymbol{\epsilon}$, we first sample its children nodes' stick weights $\{\lambda_{\boldsymbol{\epsilon}\epsilon_i} : \epsilon_i \leq N_{\boldsymbol{\epsilon}}\}$ from the conditional posterior distribution

$$\lambda_{\boldsymbol{\epsilon}\epsilon_i} = \nu_{\epsilon_i} \textstyle\prod_{i'<i}(1-\nu_{\epsilon_{i'}}),$$

$$\nu_{\epsilon_i} \sim \text{Beta}\big(1 + \sum_{m=1}^{M} 1_{\{c_m^{|\boldsymbol{\epsilon}|+1} = \boldsymbol{\epsilon}\epsilon_i\}}, \gamma + \sum_{m=1}^{M} 1_{\{c_m^{|\boldsymbol{\epsilon}|+1} > \boldsymbol{\epsilon}\epsilon_i\}}\big)$$

Then for each image m, from node $\boldsymbol{\epsilon}$ the child node $\boldsymbol{\epsilon}\epsilon^\star$ that it transits to is generated from the following proposal distribution

$$q(c_m^{|\boldsymbol{\epsilon}|+1} = \boldsymbol{\epsilon}\epsilon^\star) \propto \begin{cases} \lambda_{\boldsymbol{\epsilon}\epsilon^\star} \exp(\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon^\star}), & \epsilon^\star \leq N_{\boldsymbol{\epsilon}} \\ \lambda_{\boldsymbol{\epsilon}\epsilon^\star} \mathcal{M}_m(N_{\boldsymbol{\epsilon}}), & \epsilon^\star > N_{\boldsymbol{\epsilon}} \end{cases}$$

where $N_{\boldsymbol{\epsilon}}$ is the number of current active children nodes of $\boldsymbol{\epsilon}$, $\mathcal{M}_m(N_{\boldsymbol{\epsilon}}) = \max_{1 \leq \epsilon_i \leq N_{\boldsymbol{\epsilon}}} \{\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}\}$, and log-likelihood $\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}$ is calculated as described in Algorithm 3. The acceptance probability of the proposed child node $\boldsymbol{\epsilon}\epsilon^\star$, which we denote as $\rho_m(\boldsymbol{\epsilon}\epsilon^\star)$, is

$$\begin{cases} 1, & \epsilon^\star \leq N_{\boldsymbol{\epsilon}} \;\&\; N'_{\boldsymbol{\epsilon}} = N_{\boldsymbol{\epsilon}} \\ \min\{1, \frac{\tilde{c}_m(N_{\boldsymbol{\epsilon}})\mathcal{M}_m(N'_{\boldsymbol{\epsilon}})}{\tilde{c}_m(N'_{\boldsymbol{\epsilon}})\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}}\}, & \epsilon^\star \leq N_{\boldsymbol{\epsilon}} \;\&\; N'_{\boldsymbol{\epsilon}} < N_{\boldsymbol{\epsilon}} \\ \min\{1, \frac{\tilde{c}_m(N_{\boldsymbol{\epsilon}})\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon'_i}}{\tilde{c}_m(N'_{\boldsymbol{\epsilon}})\mathcal{M}_m(N_{\boldsymbol{\epsilon}})}\}, & \epsilon^\star_i > N_{\boldsymbol{\epsilon}} \end{cases}$$

where the normalizing constant $\tilde{c}_m(N_{\boldsymbol{\epsilon}}) = \sum_{i=1}^{N_{\boldsymbol{\epsilon}}} w_{\boldsymbol{\epsilon}\epsilon_i} \mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i} + \mathcal{M}_m(N_{\boldsymbol{\epsilon}})(1 - \sum_{i=1}^{N_{\boldsymbol{\epsilon}}} w_{\boldsymbol{\epsilon}\epsilon_i})$. Algorithm 2 summarizes the retrospective sampling scheme described above.

Algorithm 5 Retrospective sampling for parent-child node transition

Input: $\mathcal{C}_{\boldsymbol{\epsilon}}$, $N_{\boldsymbol{\epsilon}}$, $\mathcal{S}_{\boldsymbol{\epsilon}}$,
Output: $c_m^{|\boldsymbol{\epsilon}|}, \mathcal{C}_{\boldsymbol{\epsilon}}$, $N_{\boldsymbol{\epsilon}}$
for $m \in \mathcal{S}_{\boldsymbol{\epsilon}}$ **do**
Sample $w_{\boldsymbol{\epsilon}\epsilon_i}$ from conditional posterior (5.5) for $\epsilon_i \leq N_{\boldsymbol{\epsilon}}$, and from prior for $\epsilon_i > N_{\boldsymbol{\epsilon}}$
$\{\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}\}_{\epsilon_i=1}^{N_{\boldsymbol{\epsilon}}} \leftarrow$ Algorithm 3
Sample $U_i \sim \text{Uniform}[0,1]$
if $\sum_{s=1}^{\epsilon^\star-1} q_m(\boldsymbol{\epsilon}s) < U_i \leq \sum_{s=1}^{\epsilon^\star} q_m(\boldsymbol{\epsilon}s)$ **then**
Set $c_m^{|\boldsymbol{\epsilon}|+1} = \boldsymbol{\epsilon}\epsilon^\star$ with probability $\rho_m(\boldsymbol{\epsilon}\epsilon^\star)$, otherwise, leave $c_m^{|\boldsymbol{\epsilon}|+1}$ unchanged
else
$N_{\boldsymbol{\epsilon}} \leftarrow N_{\boldsymbol{\epsilon}} + 1$, $\mathcal{C}_{\boldsymbol{\epsilon}} \leftarrow \mathcal{C}_{\boldsymbol{\epsilon}} \bigcup \boldsymbol{\epsilon}\epsilon^\star$, and set $c_m^{|\boldsymbol{\epsilon}|+1} = N_{\boldsymbol{\epsilon}}$ with probability $\rho_m(\boldsymbol{\epsilon}\epsilon^\star)$, otherwise, leave $c_m^{|\boldsymbol{\epsilon}|+1}$ unchanged
end if
end for

Algorithm 6 Calculating $\{\mathcal{L}_m^{\boldsymbol{\epsilon}}\}_{\boldsymbol{\epsilon}\in\mathcal{A}}$

Input: L_m, $\{\mathcal{G}_l\}_{l=1}^{L_m}$, $\{\boldsymbol{\pi}_{\boldsymbol{\epsilon}}, w_{\boldsymbol{\epsilon}}, \mathcal{C}_{\boldsymbol{\epsilon}}\}_{\boldsymbol{\epsilon}\in\mathcal{A}}$
Output: $\{\mathcal{L}_m^{\boldsymbol{\epsilon}}\}_{\boldsymbol{\epsilon}\in\mathcal{A}}$
$l = L_m$
while $l > 1$ **do**
for $\boldsymbol{\epsilon} \in \mathcal{G}_l$ **do**

$$\hat{\mathcal{L}}_{m\boldsymbol{\epsilon}} = \sum_{i=1}^{N_m}\sum_{k=1}^{K}\left[1_{\{h_{mi}=\boldsymbol{\epsilon}\}} z_{mik}\log(\pi_{\boldsymbol{\epsilon}k}) + 1_{\{h_{mi}=\boldsymbol{\epsilon}\}}(1-z_{mik})\log(1-\pi_{\boldsymbol{\epsilon}k})\right]$$

$$\mathcal{L}_m^{\boldsymbol{\epsilon}} = \log w_{\boldsymbol{\epsilon}} + \hat{\mathcal{L}}_{m\boldsymbol{\epsilon}} + \sum_{\boldsymbol{\epsilon}\epsilon_i\in\mathcal{C}_{\boldsymbol{\epsilon}}} \mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}$$

end for
$l = l - 1$
end while

Note that after $\boldsymbol{c}_m$ is obtained for each image m, the width of the tree structure may expand as a result of the new inferred N_ϵ. Additionally, it may also contain nodes that no image is assigned to, in this case we prune the tree by deleting those empty nodes.

In Algorithm 3 we calculate $\mathcal{L}_m^{\boldsymbol{\epsilon}\epsilon_i}$ recursively from nodes at the bottom level to the top, in this way we sweep every node only once.

5.5 Experiments

We test the proposed model with five datasets: (*i*) a simulated, illustrative example that examines the ability to learn the tree structure; (*ii*) a subset of the MNIST digits data, (*iii*) face data (Tenenbaum et al., 2000); (*iv*) the Microsoft (MSRC) image database; and (*v*) the LabelMe data. In the case of (*iv*) and (*v*), the images are supplemented by annotations. For (*ii*)-(*v*), we process patches from each image. For the MNIST and face data, we randomly select 50 partially overlapping patches in each image, with 15×15 and 40×40 patch sizes, respectively (placement of patches was not tuned, selected uniformly at random, with partial overlap). For the MSRC and LabelMe data, we collect all $32 \times 32 \times 3$ non-overlapping patches from the *color* images (we also consider overlapping patches in this case, but it was found unnecessary). Recall that $\boldsymbol{x}_{mi} \in \mathbb{R}^P$, with P the number of pixels in each patch ($P = 225$ for MNIST, $P = 1600$ for the face data, and $P = 3072$ for MSRC and LabelMe data).

We have examined different methods for initializing the dictionary, including random draws from the prior and various fixed redundant bases, such as the over-complete DCT. Alternatively, we may use existing dictionary-learning methods (independent of the topic model); for this purpose, we use the covariate-dependent hierarchical beta process (with the covariates linked to the relative locations between patches) to learn an initial set of dictionary atoms (Zhou et al., 2011c). Additionally, in examples (*ii*)-(*v*), we initialize the tree with 4 levels. In this initialization, four nodes are present beneath the root node, and each subsequent node has two children, down four levels; nested K-means clustering is used to initialize the data among the clusters (nodes) at the respective levels.

In all experiments, the hyperparameters were set $a_0 = b_0 = 1$, $c_0 = d_0 = e_0 = f_0 = 10^{-6}$, $\alpha = 1$ and $\gamma = 1$, and the truncation level (upper bound) on the number

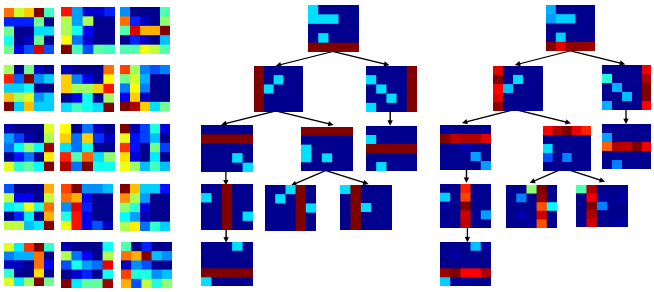


FIGURE 5.2: Example with synthesized images. Left: Example images. Middle: The ground truth for the underlying model. Right: The inferred model from the maximum-likelihood collection sample.

of dictionary elements was $K = 400$; many related settings of these parameters yield similar results, and no tuning was performed.

5.5.1 Inferring the tree: simulated data

We first illustrate that the proposed model is able to infer both the depth and width of the tree, using synthesized data for which the tree that generates the data is known; the data are like (but distinct from) that considered in Figure 2 of Blei et al. (2003a). In this simple example we wish to isolate the component of the model that infers the tree, so there is no dictionary learning. The data consists of a 25-element alphabet, arranged as 5×5 blocks on a grid; each topic is characterized by the probability of using each of the 25 elements (there is a probability π_{hk} for using element $k \in \{1, \ldots, 25\}$, for each topic/node h). For each topic h the "truth" (middle in Figure 5.2) has probabilities 0 (blue), 0.1 (cyan blue), and 0.5 (red). The generative process for image m corresponds to first drawing a path $\boldsymbol{c}_m$ through the tree, and then 1000 times a node on this patch is drawn from $\boldsymbol{\theta}_m$, and finally each of

the 25 alphabet members are drawn Bernoulli, with the associated topic-dependent probabilities (this is the proposed model, without dictionary learning). The final data ("image") consists of a count of the number of times each of the 25 elements was used, across the 1000 draws (example data at left in Figure 5.2). A total of 100 5×5 "images" were drawn in this manner to constitute the data. The right part of Figure 5.2 corresponds to the *recovered* tree, based upon the maximum-likelihood collection sample. Of course, the order of the branches and children is arbitrary; the inferred tree in Figure 5.2 (right) was arranged *a posteriori* to align with the "truth" (middle), to clarify presentation.

In this example the tree was initialized with 3 paths, each with three 3 layers (note in truth there are four paths, with variable number of layers). We also initialized the tree with 4 and 5 paths, under the same experimental setting, and similar recovery is achieved. If we initialized with less than 3 paths or more than 5, the recovered tree was still reasonable (close), but not as good. However, the inference of the topic-dependent usage probabilities $\boldsymbol{\pi}_h$ was very robust to numerous different settings. These results are based on 2000 samples, 1000 discarded as burn-in.

5.5.2 Model fit

For the MNIST handwritten digit database, we randomly choose 100 images per digit (digits 0 through 9), and therefore $M = 1000$ images are considered in total; the images are of size 28×28. The face dataset (Tenenbaum et al., 2000) contains $M = 698$ faces, each of size 64×64. Concerning the inferred trees, for the MNIST data, the maximum-likelihood collection sample had 168 paths and each path was typically 5 layers deep; for the face data 80 paths were inferred, and each was typically 5 layers. To quantitatively compare the ability of the hierarchical dictionary construction to fit the data, we consider reconstruction error for the data, comparing with the single-layer ("flat") model in Li et al. (2011); results are summarized in

Table 5.1, corresponding to $\|\boldsymbol{x}_{mi} - \mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi})\|_2^2$, averaged across all images m and patches i. In addition to results for MNIST and faces data, we show results for the MSRC and LabelMe data sets (analyzed in this case *without* annotations).

We also performed experiments to investigate how initialization affects the performance. In Table 5.1, instead of initializing the dictionary via the hierarchical beta process (hBP), they are initialized at random. While there is a slight degradation in performance with random initialization, it is not marked, and the results are still better than those produced by Li et al. (2011). Similar improvements in hBP initialization were observed for the classification task discussed below; hBP helps, but random initialization is still good.

Table 5.1: Reconstruction error comparison (mean square error multiplied by 10^3, and $\pm$ one standard deviation) on MNIST, Face, MSRC and LabelMe datasets. 'nDP+hBP' and 'nDP+random' correspond to the proposed model with the dictionary initialized by hBP and randomly, respectively, while the "flat" (single layer) model corresponds to Li et al. (2011).

	MNIST	Face	MSRC	LabelMe
flat model	11.10 $\pm$ 0.32	8.18 $\pm$ 0.17	10.27 $\pm$ 0.54	12.16 $\pm$ 0.30
nDP+hBP	10.42 $\pm$ 0.21	7.64 $\pm$ 0.12	8.64 $\pm$ 0.36	10.21 $\pm$ 0.27
nDP+random	10.85 $\pm$ 0.35	7.91 $\pm$ 0.20	9.25 $\pm$ 0.41	11.53 $\pm$ 0.50

The proposed model fits the data better than the "flat" model in Li et al. (2011); the gains are more evident when considering real and sophisticated imagery (the MSRC and LabelMe data). It is important to note that the proposed model is effectively no more complicated than the model in Li et al. (2011). Specifically, in Li et al. (2011) and in the proposed model, each image is put in a cluster, where in Li et al. (2011) a single-layer Dirichlet process was used to perform clustering, where here paths through the tree define clusters. In Li et al. (2011) and here each cluster/path is characterized by a distribution over topics, and in both models each topic is characterized by probabilities of atom usage. The difference is that in Li et al. (2011) the probabilities of topic usage for each cluster are drawn independently,

where here the tree structure, and shared nodes between different paths, manifest statistical dependencies between the probability of topic usage in different paths with shared nodes.

In these examples a total of 250 Gibbs samples were run, with 150 discarded as burn-in. The results of the model correspond to averaging across the collection samples. In all examples useful results were found with a relatively small number of Gibbs samples.

The results in Section 5.5.1 indicate that the model, with a retrospective sampler, can infer a meaningful tree, and in this section we have demonstrated that this tree can lead to better fit to the data. In the next two subsections we examine how these attributes yield better performance on the motivating application: organization and classification of images with associated annotations.

We show the full tree structure inferred from MNIST data in Figure 5.3 and full tree structure from Face data in Figure 5.4. In both hierarchies, each node is plotted as the average of all images that were assigned to it. Each image is assigned to a path of the tree. Nodes on the top level are shared by more images assigned to all the paths splitting from the node. Therefore, the averaged mean at nodes on top levels look more blurred than the one on bottom level. For the MNIST data, 8 subtopics are inferred on the second level, each representing one main type of digit. At the bottom level, much more representations of digits are found rather than 18 in (Li et al., 2011). The results are presented in a manner such that disagreements in the pose/illumination of data on the same node manifests blurriness of the average image at that node. The model captures common structure (nodes on the top layers) and idiosyncrasies (bottom nodes and leaves) characteristic of the whole dataset. The degree of similarity between two clusters (paths) is manifested by the number of nodes they share. Figure 5.5 shows the inferred dictionary atoms for both the Face data and the MSRC data considered next in a decreasing order of importance.

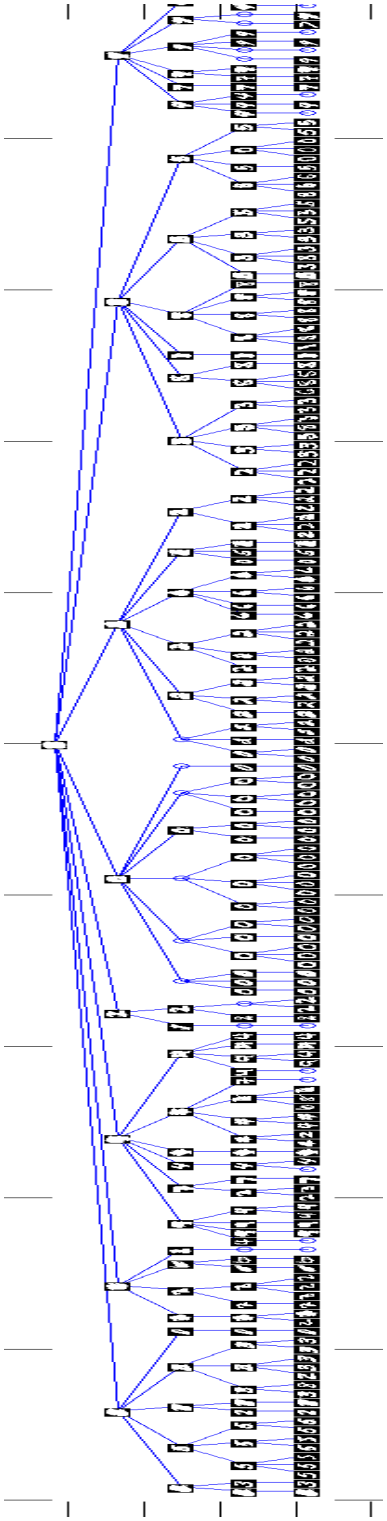


FIGURE 5.3: he full tree structure inferred from MNIST data where each node is plotted as the average of all images that were assigned to that node.

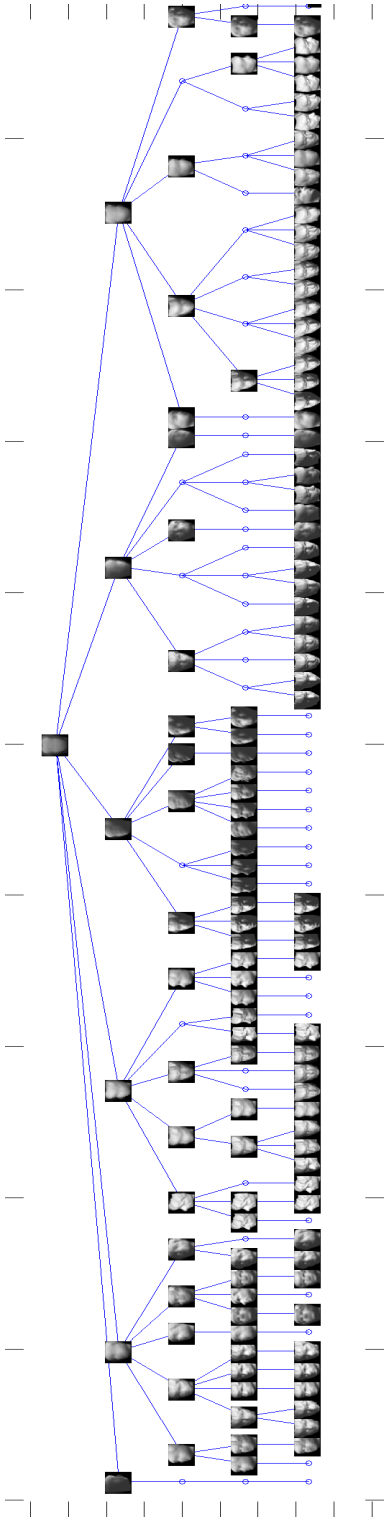


FIGURE 5.4: The full tree structure inferred from face data where each node is plotted as the average of all images that were assigned to that node.

(a) (b)

FIGURE 5.5: The inferred dictionary for Face data (a) and MSRC data (b) with elements sorted in a decreasing order of importance.

5.5.3 Organizing MSRC data

We use the same settings of images and annotations from the MSRC data[1] as considered in Du et al. (2009), to allow a direct comparison. We choose 320 images from 10 categories of images with manual annotations available. The categories are "tree", "building", "cow", "face", "car", "sheep", "flower", "sign", "book" and "chair". The numbers of images are 45 and 35 in the "cow" and "sheep" classes, respectively, and 30 in all the other classes. Each image has size 213×320 or 320×213. For annotations, we remove all words that occur less than 8 times (approximately 1% of words), and obtain 15 unique annotation-words, thus $N_v = 15$.

The full tree structure inferred is shown in Figure 5.6, with its maximum depth inferred to be 6 (maximum-likelihood collection sample depicted, from 100 collection samples). On the second level, it is clearly observed that images are clustered in several main sub-genres, *e.g.*, one with images containing grass, one with flowers, and another with urban construction, including cars and buildings, etc. For each path, we depict up to the 8 most-probable images assigned to it (fewer when less

[1] `http://research.microsoft.com/en-us/projects/objectclassrecognition/`

than 8 were assigned).

To further demonstrate the form of the model, two pairs of example images are shown in Figure 5.7. The pairs of images were assigned to two distinct paths, that shared nodes near the root (meaning the model infers shared types of patches between the images, assigned to these shared nodes). For each node, three example patches that are assigned to it are selected, from each of the two images. For the pair of example images from the "sheep" and "cow" classes, patches of grass and legs are shared on the top nodes, while distinct patches manifesting color and texture are separately assigned to nodes at bottom levels. The two images from the "building" class show the diversity of this category. It is anticipated that the "building" category will be diverse, with common patches shared at nodes near the root, and specialized patches near the leaves (for different types of buildings/structures). These typical examples illustrate that ubiquitous patches are shared at nodes near the root, with nodes toward the leaves describing details associated with specialized classes of images. It is this hierarchical structure that is missed by the model in Li et al. (2011), and that also apparently manifests the better model fit, as summarized in Table 5.1.

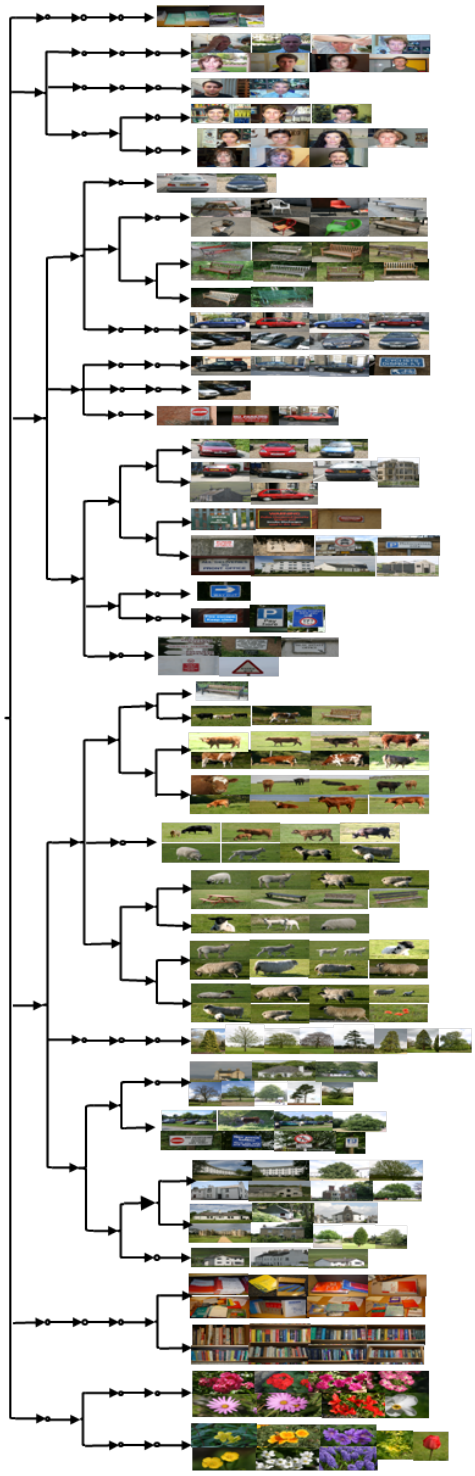


FIGURE 5.6: The full tree structure inferred from MSRC data. For each path, up to 8 images assigned to it are shown.

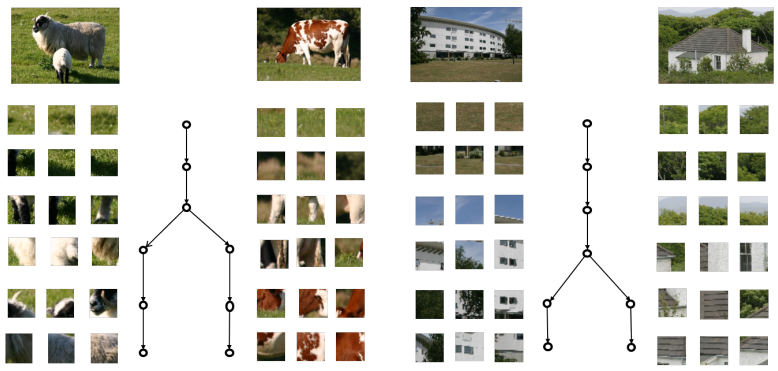


FIGURE 5.7: Two pairs of example images, and the paths they were assigned to. Between the images is shown the splitting paths. At top are the example images, and for each image we depict three example patches assigned to a respective node.

Another product of the model is a distribution over words for each path in the tree (not shown, for brevity). We illustrate this component of the model for the LabelMe data, considered next.

5.5.4 Organizing LabelMe data

The LabelMe data[2] contain 8 image classes: "coast", "forest", "highway", "inside city", "mountain", "open country", "street" and "tall building". We use the same settings of images and annotations as Wang et al. (2009): we randomly select 100 images for each class, thus the total number of images is 800. Each image is resized to be 256×256 pixels. For the annotations, we remove terms that occur less than 10 times, and obtain a vocabulary of 99 unique words, thus $N_v = 99$. There are 5 terms per annotation in the LabelMe data on average. Figure 5.10 visualizes 7 sub-trees of the inferred tree structure; there are 7 nodes inferred on the second level, and each node represents one sub-tree. Class "street" and class "insidecity" share the same root node, labeled 5 in Figure 5.10.

Based on the learned posterior word distribution $\boldsymbol{\psi}_p$ for the pth image class, we

[2] `http://www.cs.princeton.edu/~chongw/`

can further infer which words are most probable for each path. Figure 5.8 shows the ψ_p for 8 example paths (maximum-likelihood sample, from 100 collection samples), with the five largest-probability words displayed; the capital letters associated with each histogram in Figure 5.8 have associated paths through the tree as indicated in Figure 5.10. A good connection is manifested between the words and paths (examine the images and words associated with each path).

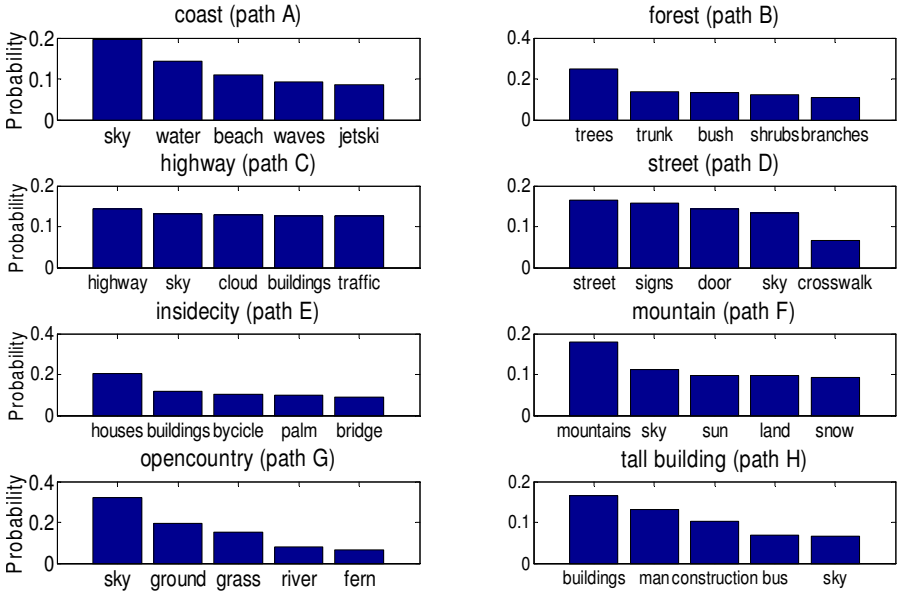
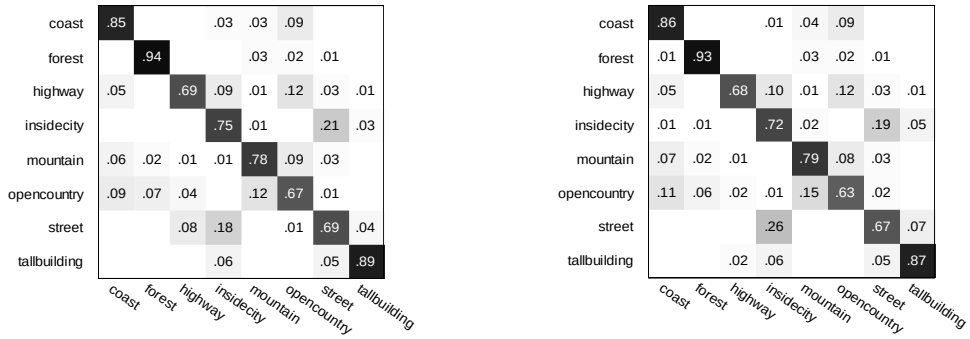


FIGURE 5.8: Inferred distributions over words for LabelMe data, as a function of inferred image category. The letters correspond to paths in Figure 5.10.

We evaluate the proposed model on the image-classification task, similar to as considered in Li et al. (2010). A set of 800 randomly selected images are held out as testing images from the 8 classes, each class with 100 testing images. Each image is represented by the estimated distribution over all the nodes in the entire hierarchy. Only nodes that are associated to the image have nonzero values in the distribution.



(a) (b)

FIGURE 5.9: For the 800 testing images from LabelMe data, (a): Confusion Matrix on original patches with the average accuracy of 78.3%. (b): Confusion Matrix on SIFT features with the average accuracy of 76.9%.

We calculate the χ^2-distances between the node distribution of the testing images and those of the training images. The KNN algorithm (K is set to be 50) is then applied to obtain the class label. Figure 5.9(a) shows the confusion matrix of classification, with an average classification accuracy of 78.3%, compared with 76% in Li et al. (2011).

We now compare the proposed hierarchical model with the hierarchical model in Li et al. (2010), in which offline SIFT feature extraction and VQ are employed. Based on related work in Wang et al. (2009), we used a codebook of size 240, and achieved an average classification accuracy of 77.4%, compared with 79.6% reported above for our algorithm. Note, however, that we found the model in Li et al. (2010) to be very sensitive to the codebook size, with serious degradation in performance manifested with 150 or 400 codes, for example. To further test the proposed model, we considered the same classification experiment on MSRC data, which is characterized by 10 classes. Five images per class were randomly chosen as testing data, and the remaining images are treated as training data to learn a hierarchical structure. An average accuracy of 64% is obtained with the proposed model, compared with 60% using that in Li et al. (2010), where the codebook size is set to be 200. These

experiments indicate that the proposed model typically does better than that in Li et al. (2010) for the classification task, even when we optimize the latter with respect to the number of codes (for which the model in Li et al. (2010) is sensitive).

In all of the above examples the dictionary learning was applied directly to the observed pixel values within a given patch, with no *a priori* feature extraction. Alternatively, the patch-dependent data $\boldsymbol{x}_{mi}$ may correspond to features extracted using *any* image feature extraction algorithm. To illustrate this, we now let $\boldsymbol{x}_{mi}$ correspond to SIFT features (Lowe, 1999) on the same patches; in this experiment the dictionary learning replaces the VQ step in models like Wang et al. (2009). In Figure 5.9(b) we show the confusion matrix of the model based on SIFT features, with an average accuracy of 76.9%, slightly better than the results reported in Wang et al. (2009) (but here there is no need to tune the number of VQ codes). This also demonstrates that performing dictionary learning directly on the patches, rather than via a state-of-the-art feature extraction method, yields highly competitive results.

The experiments above have been performed in 64-bit Matlab on a machine with 2.27 GHz CPU and 4 Gbyte RAM. One MCMC sample of the proposed model takes approximately 4, 2, 8 and 10 minutes respectively for the MNIST, Face, MSRC and LabelMe experiments (in which we simultaneously analyzed respectively 1000, 698, 320, and 800 total images). Note that while these model *learning* times are relatively expensive, model testing (after the tree and dictionary are learned) is very fast, this employed for the aforementioned classification task. We have focused here on model development and demonstration, not on optimizing implementation. To scale the model up to larger numbers of training images, we may perform variational Bayesian inference rather than sampling, and employ online-learning methods (Honkela and Valpola, 2003). One may also consider online sampling methods, such as particle filters and related methods (Lopes and Carvalho, 2013).

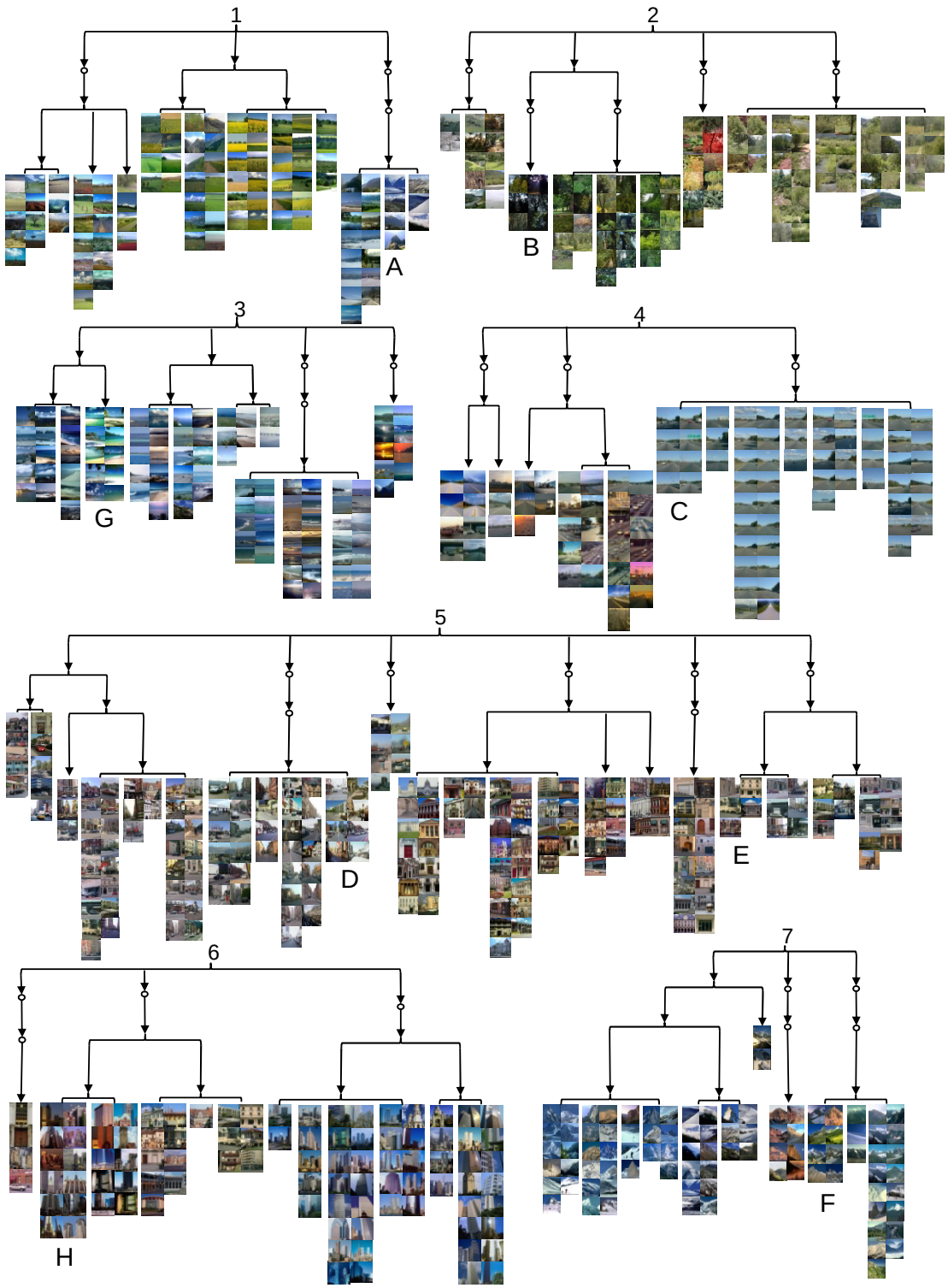


FIGURE 5.10: The structure of 7 sub-trees inferred from LabelMe data. The root of each sub-tree is one child node of the root for the entire tree structure. For each path, all images assigned to it are listed with no order importance. The letters refer to paths for which distributions over words are depicted in Figure 5.8.

5.6 Summary

The nested Dirichlet process has been integrated with dictionary learning to constitute a new hierarchical topic model for imagery. The dictionary learning may be employed on the original image pixels, or on features from any image feature extractor. If words are available, they may be utilized as well, with word-dependent usage probabilities inferred for each path through the tree. A retrospective sampling method has been developed to infer both the tree depth and width. Encouraging qualitative and quantitative results have been demonstrated for analysis of many of the traditional datasets in this field, with comparisons provided to other related published methods.

6

Conclusion and Future work

In this thesis, we have introduced several novel hierarchical Bayesian models based on dictionary learning for imagery and text analysis, with the following contributions:

1. The nonparametric beta process factor analysis (BPFA) model with the probit stick-breaking process (PSBP) prior is developed for image interpretation and image segmentation simultaneously with efficient Gibbs Sampling inference. The proximate patches are supposed to share similar probabilities over dictionary atoms, thereby encouraged to be clustered within the same segment type. The use of a probit link function to define space dependent stick weights yields contiguous segments with sharp boundaries.

2. An online variational Bayesian (VB) algorithm for dictionary learning is developed for large-scale datasets. This online algorithm is based on online stochastic optimization with a natural gradient step, which guarantees the algorithm to converge to a local optimal of the VB objective function. Furthermore, a parallel Map-Reduce framework implemented on the hadoop server is developed.

3. A novel nonparametric Bayesian model to integrate dictionary learning and topic modeling into a unified framework is proposed to organize a collection of par-

tially annotated images. Feature extraction is no longer separated from image clustering part, but instead is employed by performing dictionary learning directly on original image patches, joint with topic modeling. Efficient inference is performed with a Gibbs-slice sampler, and encouraging results are achieved by comparing with a conventional image clustering analysis.

4. A tree-based dictionary learning model is developed for joint analysis of imagery and associated text. The dictionary learning may be applied directly to the imagery from patches, or to general feature vectors extracted from patches or superpixels (using any existing method for image feature extraction). Each image is associated with a path through the tree (from root to a leaf), and each of the multiple patches in a given image is associated with one node in that path. Nodes near the tree root are shared between multiple paths, representing image characteristics that are common among different types of images. Moving toward the leaves, nodes be- come specialized, representing details in image classes. If available, words (text) are also jointly modeled, with a path-dependent probability over words. The tree structure is inferred via a nested Dirichlet process, and a retrospective stick-breaking sampler is used to infer the tree depth and width.

These contributions motivate several directions for future work:

1. Extend a hierarchical model for joint segmenting and reconstructing multiple images. In Chapter 2, we focused on single image segmentation and inpainting, but this could be naturally extended to *multiple* images, by sharing all the segment components. Motivated by the hierarchical Dirichlet process (HDP) (Teh et al., 2004b), each image is associated with a PSBP prior over latent segment components, and these components from all images share the same base measure, which is drawn from a DP above. Besides, the dictionary atoms representing image features and hidden structures will be shared across all images. The segmentation of each image could further be utilized to sort, search and classify images, borrowing the idea from

Chapter 4.

2. Exploit online learning or parallel framework for joint imagery and text analysis in both flat and hierarchical ways. In real world, it is desired to organize large-scale image datasets. Our current algorithms reported in Chapter 4 and 5 are both based on batch learning, which is limited to the size of datasets. Similar techniques as in Chapter 3 could be taken to develop an online learning algorithm for the proposed models so as to handle massive datasets. This work should be very attractive especially in the current "big-data" world.

3. Exploit image classification problem via dictionary learning. As in the model discussed in Chapter 4, each image could be represented by a histogram of topics which each patch is assigned to. Motivated by (Blei and MaAuliffe, 2007), given the label information in the training set, classifiers could be trained for each category over these histograms. We could further consider the spatial information among patches as in Chapter 2 to impose dependence over topics. With the help of annotations, classification accuracy should also be improved.

4. Motived by (Zhou et al., 2010), the proposed parallel and online learning algorithms of BPFA model reported in Chapter 3 could naturally be extended for collaborative filtering applications such as Netflix award or large-scale recommendation system. Under Bayesian framework, this proposed model could easily incorporate user or/and movie features as well as the user or movie bias term. Furthermore, this model is robust to random initialization and noisy assumption. The implemented parallel version could be very powerful and flexible for large-scale recommendation system with state-of-the-art performance.

Appendix A

Posterior Update Equations for BPFA_PSBP model

The Gibbs sampling inference for the model variables are summarized as follows:

1) Sample $\boldsymbol{d}_k$:

$$p(\boldsymbol{d}_k|\sim) \propto \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\Sigma}_i\mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i), \gamma_\epsilon^{-1}\mathbf{I}_{||\boldsymbol{\Sigma}_i||_0})\mathcal{N}(\boldsymbol{d}_k|0, P^{-1}\mathbf{I}_P) \propto \mathcal{N}(\boldsymbol{d}_k|\boldsymbol{\mu}_{\boldsymbol{d}_k}, \boldsymbol{\Sigma}_{\boldsymbol{d}_k}) \tag{A.1}$$

with the covariance $\boldsymbol{\Sigma}_{\boldsymbol{d}_k}$ and the mean $\boldsymbol{\mu}_{\boldsymbol{d}_k}$ expressed as

$$\boldsymbol{\Sigma}_{\boldsymbol{d}_k} = \left(P\mathbf{I} + \gamma_\epsilon \sum_{i=1}^{N} z_{ik}^2 s_{ik}^2 \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i\right)^{-1} \tag{A.2}$$

$$\boldsymbol{\mu}_{\boldsymbol{d}_k} = \gamma_\epsilon \boldsymbol{\Sigma}_{\boldsymbol{d}_k} \sum_{i=1}^{N} z_{ik} s_{ik} \tilde{x}_i^{-k} \tag{A.3}$$

where

$$\tilde{x}_i^{-k} = \boldsymbol{\Sigma}_i^T \boldsymbol{y}_i - \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i) + \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \boldsymbol{d}_k (s_{ik} \circ z_{ik}) \tag{A.4}$$

2) Sample $\boldsymbol{s}_k = (s_{i1}, ..., s_{Nk})$:

$$p(\boldsymbol{s}_i|\sim) \propto \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\Sigma}_i\mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i), \gamma_\epsilon^{-1}\mathbf{I}_{||\boldsymbol{\Sigma}_i||_0})\mathcal{N}(\boldsymbol{s}_i|0, \gamma_s^{-1}\mathbf{I}_K) \tag{A.5}$$

It can be shown that s_{ik} can be drawn from a normal distribution $p(s_{ik}|\sim) \sim \mathcal{N}(s_{ik}|\mu_{s_{ik}}, \Sigma_{s_{ik}})$ with the variance $\Sigma_{s_{ik}}$ and the mean $\mu_{s_{ik}}$ express as

$$\Sigma_{s_{ik}} = (\gamma_s + \gamma_\epsilon z_{ik}^2 \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \boldsymbol{d}_k)^{-1} \tag{A.6}$$

$$\mu_{s_{ik}} = \gamma_s \Sigma_{s_{ik}} z_{ik} \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \tilde{\boldsymbol{x}}_i^{-K} \tag{A.7}$$

3) Sample $\boldsymbol{z}_{:k} = (z_{1k}, ..., z_{Nk})$

$$p(z_{ik}|\sim) \propto \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{||\boldsymbol{\Sigma}_i||_0}) \text{Bernoulli}(z_{ik}|0, \pi_{h_i k}) \tag{A.8}$$

The posterior probability that $z_{ik} = 1$ is proportional to

$$p1 = \pi_{h_i k} \exp(-\frac{\gamma_\epsilon}{2}(s_{ik}^2 \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \boldsymbol{d}_k - 2 s_{ik} \boldsymbol{d}_k^T \tilde{\boldsymbol{x}}_i^{-k})) \tag{A.9}$$

and the posterior probability that $z_{ik} = 0$ is proportional to

$$p0 = 1 - \pi_{h_i k} \tag{A.10}$$

so z_{ik} can be drawn from a bernoulli distribution as

$$z_{ik} \sim \text{Bernoulli}(\frac{p1}{p0 + p1}) \tag{A.11}$$

4) Sample $\boldsymbol{\pi}_j$ for $j = 1, ..., J$: the dictionary usage for layer j is sampled from a beta distribution as:

$$p(\boldsymbol{\pi}_j|\sim) \sim \text{Beta}(\boldsymbol{a}_j, \boldsymbol{b}_j) \tag{A.12}$$

where

$$\boldsymbol{a}_j = a_0/K + \sum_{i=1}^{N} \delta(h_i = j) \boldsymbol{z}_i \tag{A.13}$$

$$\boldsymbol{b}_j = b_0(K-1)/K + \sum_{i=1}^{N} \delta(h_i = j)(1 - \boldsymbol{z}_i) \tag{A.14}$$

5) Sample $\boldsymbol{\nu}_i = (\nu_{i1}, ..., \nu_{iJ})^T$

$$p(\nu_{i1}, ..., \nu_{iJ}) \propto \prod_{j=1}^{J} p(\nu_{ij}|\boldsymbol{\zeta}_j) \times p(\boldsymbol{z}_i|\{\nu_{ij}, \boldsymbol{\pi}_j\}_{j=1}^{J}) \tag{A.15}$$

$$= \prod_{j=1}^{J} p(\nu_{ij}|\boldsymbol{\zeta}_j) \times \prod_{j=1}^{J} p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\phi_{ij}}$$

$$= \prod_{j=1}^{J} p(\nu_{ij}|\boldsymbol{\zeta}_j) \times \prod_{j=1}^{J} p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\nu_{ij}\prod_{j'=1}^{j-1}(1-\nu_{ij'})}$$

$$= \prod_{j=1}^{J} p(\nu_{ij}|\boldsymbol{\zeta}_j) \times$$

$$p(\boldsymbol{z}_i|\boldsymbol{\pi}_1)^{\nu_{i1}} p(\boldsymbol{z}_i|\boldsymbol{\pi}_2)^{\nu_{i2}(1-\nu_{i1})} \dots p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\nu_{ij}\prod_{j'=1}^{j-1}(1-\nu_{ij'})} \dots p(\boldsymbol{z}_i|\boldsymbol{\pi}_J)^{\nu_{iJ}\prod_{j'=1}^{J-1}(1-\nu_{ij'})}$$

From (A), it is easy to have

$$p(\nu_{ij}| \sim) \propto p(\nu_{ij}|\boldsymbol{\zeta}_j) \times [p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\nu_{ij}\prod_{j'=1}^{j-1}(1-\nu_{ij'})}] \times [\prod_{\hat{j}>j} p(\boldsymbol{z}_i|\boldsymbol{\pi}_{\hat{j}})^{(1-\nu_{ij})\nu_{i\hat{j}}\prod_{l<\hat{j},l\neq j}(1-\nu_{il})}] \tag{A.16}$$

Therefore, we have

$$p(\nu_{ij} = 1| \sim) \propto p(\nu_{ij} = 1|\boldsymbol{\zeta}_j) p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\prod_{j'<j}(1-\nu_{ij'})} \tag{A.17}$$

$$= \Phi(g_j(\boldsymbol{l}_i)) p(\boldsymbol{z}_i|\boldsymbol{\pi}_j)^{\prod_{j'<j}(1-\nu_{ij'})}$$

$$p(\nu_{ij} = 0| \sim) \propto p(\nu_{ij} = 0|\boldsymbol{\zeta}_j) \prod_{l>j} p(\boldsymbol{z}_i|\boldsymbol{\pi}_l)^{\nu_{il}\prod_{j'<l,j'\neq j}(1-\nu_{ij'})} \tag{A.18}$$

$$= [1 - \Phi(g_j(\boldsymbol{l}_i))] \prod_{l>j} p(\boldsymbol{z}_i|\boldsymbol{\pi}_l)^{\nu_{il}\prod_{j'<l,j'\neq j}(1-\nu_{ij'})}$$

For computational consideration, we let

$$r = \log \frac{p(\nu_{ij} = 1| \ )}{p(\nu_{ij} = 0| \ )} \tag{A.19}$$

$$= \log \frac{\Phi(g_j(\boldsymbol{l}_i))}{1 - \Phi(g_j(\boldsymbol{l}_i))} + \prod_{j'<j}(1 - \nu_{ij'}) \log p(\boldsymbol{z}_i|\boldsymbol{\pi}_j) - \sum_{j'>j}[\nu_{ij'} \prod_{l<j',l\neq j'} (1 - \nu_{il})] \log p(\boldsymbol{z}_i|\boldsymbol{\pi}'_j)$$

where

$$\log p(\boldsymbol{z}_i|\boldsymbol{\pi}_j) = \sum_{k=1}^{K}[z_{ik}\log\pi_{jk} + (1-z_{ik})\log(1-\pi_{jk})] \tag{A.20}$$

Thus, we could easily get

$$p(\nu_{ij}=1|\sim) = 1 - \frac{1}{1+e^r} \tag{A.21}$$

$$p(\nu_{ij}=0|\sim) = \frac{1}{1+e^r} \tag{A.22}$$

6)Sample $\boldsymbol{\nu}_i^* = (\nu_{i1}^*, \ldots, \nu_{iJ}^*)$:

$$\begin{aligned} p(\nu_{ij}^*|\sim) &\propto p(\nu_{ij}^*|\boldsymbol{\zeta}_j)p(\nu_{ij}|\nu_{ij}^*) \\ &\propto \mathcal{N}(\nu_{ij}^*|g_j(\boldsymbol{l}_i),1)[1(\nu_{ij}^*>0)\delta_1(\nu_{ij}) + 0(\nu_{ij}^*<0)\delta_0(\nu_{ij})] \\ &= \mathcal{N}_+(\nu_{ij}^*|g_j(\boldsymbol{l}_i),1)\delta_1(\nu_{ij}) + \mathcal{N}_-(\nu_{ij}^*|g_j(\boldsymbol{l}_i),1)\delta_0(\nu_{ij}) \end{aligned} \tag{A.23}$$

7) Sample $\boldsymbol{\omega}_j$:

$$\begin{aligned} p(\boldsymbol{\omega}_j|\sim) &\propto p(\boldsymbol{\omega}_j)\prod_{i=1}^{N}p(\nu_{ij}^*|\boldsymbol{\omega}_j) = \mathcal{N}(\boldsymbol{\omega}_j;0,\lambda_j^{-1})\prod_{i=1}^{N}\mathcal{N}(\nu_{ij}^*|\boldsymbol{\omega}_j^T\Phi_i^j,1) \\ &\propto \mathcal{N}(\boldsymbol{\omega}_j|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) \end{aligned} \tag{A.24}$$

It can be shown that the posterior of the covariance $\boldsymbol{\Sigma}_j$ and the mean $\boldsymbol{\mu}_j$ could be expressed as

$$\boldsymbol{\Sigma}_j = (\lambda_j + \sum_{i=1}^{N}\Phi_i^j\Phi_i^{j^T})^{-1} \tag{A.25}$$

$$\boldsymbol{\mu}_j = \boldsymbol{\Sigma}_j\sum_{i=1}^{N}\nu_{ij}^*\Phi_i^j \tag{A.26}$$

where $\boldsymbol{\Phi}$ is a $N \times N_c + 1$ 'design' matrix where $\Phi_{in}^j = \mathcal{K}(\boldsymbol{l}_i, \hat{\boldsymbol{l}}_n; \psi_j)$

8)Sample λ_j:

$$p(\lambda_j|\sim) \propto \mathrm{Ga}(\lambda_j|\alpha_{01},\alpha_{02})\mathcal{N}(\omega_j|0,\lambda_j^{-1}) \sim \mathrm{Ga}(\lambda_j|\tau_1,\tau_2) \tag{A.27}$$

where

$$\tau_1 = \tau_{01} + \frac{1}{2} \tag{A.28}$$

$$\tau_2 = \tau_{02} + \frac{1}{2}\zeta_{ij}^2 \tag{A.29}$$

9) Sample γ_s:

$$p(\gamma_s| \sim) \propto \Gamma(\gamma_s|c_0, d_0) \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{s}_i|0, \gamma_s^{-1}\mathbf{I}_K) \tag{A.30}$$

It can be shown that γ_s can be drawn from a Gamma distribution as

$$p(\gamma_s| \sim) \sim \text{Ga}(c_0 + \frac{1}{2}KN, d_0 + \frac{1}{2}\sum_{i=1}^{N} \boldsymbol{s}_i^T \boldsymbol{s}_i) \tag{A.31}$$

10) Sample γ_ϵ:

$$p(\gamma_\epsilon| \sim) \propto \Gamma(\gamma_\epsilon; e_0, f_0) \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\Sigma}_i\mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i), \gamma_\epsilon^{-1}\mathbf{I}_{||\boldsymbol{\Sigma}_i||_0}) \tag{A.32}$$

It can be shown that γ_s can be drawn from a Gamma distribution as

$$p(\gamma_\epsilon| \sim) \sim \text{Ga}(e_0 + \frac{1}{2}\sum_{i=1}^{N} ||\boldsymbol{\Sigma}_i||_0, f_0 + \frac{1}{2}\sum_{i=1}^{N} ||\tilde{\boldsymbol{x}}_i^{-k}||_{l_2}) \tag{A.33}$$

Note that $\boldsymbol{\Sigma}_i^T\boldsymbol{\Sigma}_i$ is a sparse identity matrix, $\boldsymbol{\Sigma}_{\boldsymbol{d}_k}$ is a diagonal matrix, and $\mathbf{Z}$ is a sparse matrix, it is easy to find that only basic arithmetical operations are needed and many unnecessary calculations can be avoided, leading to fast computation and low memory requirement.

Appendix B

Supplement to Nested Dictionary Learning for Hierarchical Organization of Imagery and Text

B.1 Nested K-means

The nested K-means is only taken as the initialization step, as a convenient a simple way of initially assigning images to prospective nodes in the tree. At the beginning of inference, we need to initialize parameters of nodes by assigning images to them. For each node, we take a K-means to assign images living at this node to all its children nodes, where K is the number of its children nodes. For example, as mentioned in the paper, four nodes are present beneath the root node (at initialization). We take the K-means algorithm where K equals 4 and assign all the images to one of the four children nodes. Then for each node at the second level, we again take the K-means algorithm where K equals 2 to assign those images assigned at this parent node to its two children nodes. This process stops when the bottom level is reached. This is what we call nested K-means. Since this nested K-means algorithm is only taken for initialization, we set relatively smaller K for each node. After the initialization, our model automatically learns the right size of the tree, including the number of

Table B.1: Symbol description of the variables

Symbol	Description
$\boldsymbol{x}_{mi}$	the ith patch in image m
$\boldsymbol{y}_{mn}$	the nth annotation word associated with image m
$\boldsymbol{z}_{mi}$	the K-dimensional binary vector for the ith patch in image m
$\boldsymbol{s}_{mi}$	the K-dimensional vector of real weights for the ith patch in image m
$\boldsymbol{d}_k$	the P-dimensional vector of dictionary atom k
$\boldsymbol{\pi}_{\boldsymbol{\epsilon}}$	the K-dimensional vector probability of dictionary for node $\boldsymbol{\epsilon}$
h_{mi}	node index for patch i in image m
$\boldsymbol{\theta}_m$	topic distributions for image m
$\boldsymbol{\psi}_p$	topic distribution over vocabulary
$c_m^{\lvert\boldsymbol{\epsilon}\rvert}$	node index that image m chooses at level $\lvert\boldsymbol{\epsilon}\rvert$
$\mathcal{A}$	the set of usable nodes in the infinite tree
l_{mi}	level allocation index for patch i in image m
l'_{mi}	the proposed level allocation index for patch i in image m
$L_m = \max_i\{l_{mi}\}$	the depth of the path that image m assigned to
$L'_m = \max_i\{l'_{mi}\}$	the depth of the path that image m assigned to when replacing l_{mi} with the proposed l'_{mi}
$\mathcal{S}_{\boldsymbol{\epsilon}} = \{m : c_m^{\lvert\boldsymbol{\epsilon}\rvert} = \boldsymbol{\epsilon}\}$	the set of images whose assignments include node $\boldsymbol{\epsilon}$
$\mathcal{C}_{\boldsymbol{\epsilon}} = \{\boldsymbol{\epsilon}\epsilon_i, i = 1, 2, \dots\}$	set of $\boldsymbol{\epsilon}$'s children nodes
$N_{\boldsymbol{\epsilon}} = \lvert C_{\boldsymbol{\epsilon}}\rvert$	number of $\boldsymbol{\epsilon}$'s children nodes
$N'_{\boldsymbol{\epsilon}}$	number of $\boldsymbol{\epsilon}$'s children nodes after replacing $c_m^{\lvert\boldsymbol{\epsilon}\rvert+1}$ with $\boldsymbol{\epsilon}\epsilon^\star$
$\mathcal{L}_m^{\boldsymbol{\epsilon}}$	log-posterior for image m taking node $\boldsymbol{\epsilon}$
$\hat{\mathcal{L}}_{m\boldsymbol{\epsilon}}$	log-likelihood for image m taking node $\boldsymbol{\epsilon}$
$\mathcal{G}_l$	set of nodes at level l

children nodes for each node.

B.2 Supplementary Inference

Table B.1 summarizes the notations of variables used. Beyond sampling $\{l_{mi}, \boldsymbol{c}_m\}$, as discussed above, we provide update equations for other parameters in this model:

- Sample $\boldsymbol{d}_k$

  The posterior of the kth dictionary atom $\boldsymbol{d}_k$ can be shown to be normal with

covariance $\mathbf{\Sigma}_{\boldsymbol{d}_k}$ and mean $\boldsymbol{\mu}_{\boldsymbol{d}_k}$

$$\mathbf{\Sigma}_{\boldsymbol{d}_k} = (P + \gamma_\epsilon \sum_{m=1}^{M} \sum_{i=1}^{N_m} z_{mik}^2 s_{mik}^2)^{-1} \tag{B.1}$$

$$\boldsymbol{\mu}_{\boldsymbol{d}_k} = \gamma_\epsilon \mathbf{\Sigma}_{\boldsymbol{d}_k} \sum_{m=1}^{M} \sum_{i=1}^{N_m} \widetilde{\boldsymbol{x}}_{mi}^{-k} s_{mik} s_{mik} \tag{B.2}$$

where $\widetilde{\boldsymbol{x}}_{mi}^{-k} = \boldsymbol{x}_{mi} - \mathbf{D}(\boldsymbol{z}_{mi} \odot \boldsymbol{s}_{mi}) + \boldsymbol{d}_k(z_{mik} \odot s_{mik})$.

- Sample z_{mik}

  For the ith patch in the mth image, sample the binary sparse code $\boldsymbol{z}_{mi} = (z_{mi1}, ..., z_{miK})$ with

$$P(z_{mik} = 1) = \pi_{h_{mik}} \exp[-\frac{\gamma_\epsilon}{2}(\boldsymbol{d}_k^T \boldsymbol{d}_k s_{mik}^2 - 2\boldsymbol{d}_k^T s_{mik} \widetilde{\boldsymbol{x}}_{mi}^{-k})] \tag{B.3}$$

$$P(z_{mik} = 0) = 1 - \pi_{h_{mik}} \tag{B.4}$$

  Thus whether the kth dictionary atom will be chosen for the ith patch of the mth image is drawn $z_{mik} \sim \text{Bernoulli}(\frac{P(z_{mik}=1)}{P(z_{mik}=1)+P(z_{mik}=0)})$.

- Sample s_{mik}

  The posterior of positive weight s_{mik} can be obtained as a truncated normal distribution, with covariance $\mathbf{\Sigma}_{s_{mik}}$ and mean $\boldsymbol{\mu}_{s_{mik}}$, where

$$\mathbf{\Sigma}_{s_{mik}} = 1/(\gamma_s + \gamma_\epsilon \boldsymbol{d}_k^T \boldsymbol{d}_k z_{mik}^2) \tag{B.5}$$

$$\boldsymbol{\mu}_{s_{mik}} = \gamma_\epsilon \mathbf{\Sigma}_{s_{mik}} \boldsymbol{d}_k^T z_{mik} \widetilde{\boldsymbol{x}}_{mi}^{-k} \tag{B.6}$$

- Sample $\boldsymbol{\pi}_{\boldsymbol{\epsilon}}$

$$\pi_{\boldsymbol{\epsilon}k} \sim \text{Beta}\big(\frac{a_0}{K} + \sum_{m=1}^{M} \sum_{i=1}^{N_m} 1\{h_{mi} = \boldsymbol{\epsilon}\} z_{mik}, \frac{b_0(K-1)}{K} + \tag{B.7}$$

$$\sum_{m=1}^{M} \sum_{i=1}^{N_m} 1\{h_{mi} = \boldsymbol{\epsilon}\}(1 - z_{mik})\big) \tag{B.8}$$

- Sample $\boldsymbol{\theta}_m$

  When sampling $\boldsymbol{\theta}_m$, assignment $\boldsymbol{c}_m$ is fixed. For $\boldsymbol{\epsilon} \in \boldsymbol{c}_m$, we first sample

$$\mu_{m|\boldsymbol{\epsilon}|} \sim \text{Beta}(1 + \sum_{i=1}^{N_m} 1\{l_{mi} = |\boldsymbol{\epsilon}|\}, \alpha + \sum_{i=1}^{N_m} 1\{l_{mi} > |\boldsymbol{\epsilon}|\}) \tag{B.9}$$

  and then construct $\theta_{m|\boldsymbol{\epsilon}|} = \mu_{m|\boldsymbol{\epsilon}|} \prod_{|\boldsymbol{\epsilon}'|<|\boldsymbol{\epsilon}|} (1 - \mu_{m|\boldsymbol{\epsilon}'|})$.

- Sample $\boldsymbol{\psi}_p$

  Posterior of $\boldsymbol{\psi}_p$ is still Dirichlet distributed as $\boldsymbol{\psi}_p \sim \text{Dir}(\zeta_{p1}, \ldots, \zeta_{pN_v})$ where

$$\zeta_{pv} = \sum_{m:\boldsymbol{c}_m=p} \sum_{n=1}^{W_m} 1\{y_{mn} = v\} \tag{B.10}$$

Bibliography

R. Adams, Z. Ghahramani, and M. Jordan. Tree-structured stick breaking for hierarchical data. In *NIPS*, 2010.

M. Aharon, M. Elad, and A. M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54:4311–4322, 2006.

D. Aldous. Exchangeability and related topics. *Ecole d'Ete de Probabilities de Saint-Flour XIII*, pages 1–198, 1985.

S. Amari. Natual gradient works effeciently in learning. *Neural Computation*, pages 10(2):251–276, 1998.

Q. An, C. Wang, I. Shterev, E. Wang, L. Carin, and D. B. Dunson. Hierarchical kernel Stickbreaking process for multi-task image analysis. In *Proceedings of the International Conference of Machine Learning*, 2008.

C.E. Antoniak. Mixtures of dirichlet processes with applications to nonparametric mixtures. *Annals of Statistics*, page 1152?1174, 1974.

H. Attias. A variational Bayesian framework for graphical models. *NIPS*, 2000.

K. Barnard, P. Duygulu, D. Forsyth, N. Freitas, D. Blei, and M. Jordan. Matching words and pictures. *JMLR*, 2003.

D. Blackwell and J. B. MacQueen. Ferguson distributions via polya urn schemes. *Annals of Statistics*, pages 1:353–355, 1973.

D. Blei, T. Griffiths, and M. Jordan. The chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 2010.

D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*, 2003a.

D. Blei and M. Jordan. Modeling annotated data. In *SIGIR*, 2003.

D. Blei and M. Jordan. Variational methods for the dirichlet process. In *International Conference on Machine Learning*, 2004.

D. Blei and J. MaAuliffe. Supervised topic models. In *NIPS*, 2007.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003b. ISSN 1532-4435.

D.M. Blei and J.D. Lafferty. Dynamic topic models. In *Proceedings of the International Conference on Machine Learning*, 2006.

Lon Bottou. Online learning and stochastic approximations, 1998.

E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inform. Theory*, 2010.

E.J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 2010.

S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 1999.

Y. Chung and D.B. Dunson. Nonparametric bayes conditional distribution modeling with variable selection. *Journal of the American Statistical Association*, 104:1646–1660, 2009.

J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. pages 107–113, 2008.

L. Du, L. Ren, D. Dunson, and L. Carin. Bayesian model for simultaneous image clustering, annotation and object segmentation. In *NIPS*, 2009.

J. A. Duan, M. Guindani, and A. E. Gelfand. Generalized spatial dirichlet process models. 2007a.

J. A. Duan, M. Guindani, and A. E. Gelfand. Generalized spatial dirichlet process models. *Biometrika*, pages 94(4):809–825, 2007b.

D. B. Dunson and J. Park. Kernel stick-breaking processes. *Biometrika*, page 95:307C323, 2007.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE TIP*, 2006.

C. E.Rasmussen. The infinite gaussian mixture model. In *Proc. Neural Information Processing Systems*, 2000.

L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

T. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973a.

T. S. Ferguson. A bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1973b.

E. B. Fox, E.B. Sudderth, and A. S. Willlsky. Hierarchical dirichlet processes for tracking maneuvering targets. In *International Conference on Information Fusion*, 2007.

W.R. Gilks, G. O. Roberts, and S. K. Sahu. Adaptive markov chain monte carlo through regeneration. *J. Amer. Statist. Assoc.*, 93:1045–1054, 1998.

T. L. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *NIPS*, 2005a.

T.L. Griffiths and Z. Ghahramani. Infinite latent feature models and the Indian buffet process. In *Proc. Advances in Neural Information Processing Systems*, pages 475–482, 2005b.

M.D. Hoffman, D.M. Blei, and F. Bach. Online learning for latent Dirichlet allocation. *NIPS*, 2010.

A. Honkela and H. Valpola. Online variational bayesian learning. In *ICA*, 2003.

H. Ishwaran and L. James. Gibbs sampling methods for stick-breaking priors. *J. Amer. Stat. Assoc*, 2001.

T. Iwata, T. Yamada, and N. Ueda. Modeling social annotation data with content relevance using a topic model. In *NIPS*, 2009.

J.F.C. Kingman. Completely random measures. *Pacific Journal of Mathematics*, page 21(1):59?78, 1967.

D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In *Proc. International Conference on Independent Component Analysis and Signal Separation*, 2007.

N.D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proc. International Conference on Machine Learning*, pages 601–608, 2009.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

H. Lee, A. Battle, R. Raina, and A.Y. Ng. Efficient sparse coding algorithms. *NIPS*, 2007.

L. Li, M. Zhou, G. Sapiro, and L. Carin. On the integration of topic modeling and dictionary learning. In *ICML*, 2011.

L.-J. Li and L. Fei-Fei. What, where and who? classifying events by scene and object recognition. In *ICCV*, 2007.

L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding: classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009.

Li-Jia Li, Chong Wang, Yongwhan Lim, David Blei, and Li Fei-Fei. Building and using a semantivisual image hierarchy. In *CVPR*, 2010.

H. F. Lopes and C. M. Carvalho. Online bayesian learning in dynamic models: An illustrative introduction to particle methods. *Hierarchical Models and Markov Chain Monte Carlo*, 2013.

D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

S. N. MacEachern. Dependent nonparametric processes. In *ASA Proceedings of the Section on Bayesian Statistical Science*, 1999.

S. N. MacEachern and P. Muller. Estimating mixture of dirichlet process models. *Journal of Computational and Graphical Statistics*, pages 223–238, 1998.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proc. International Conference on Machine Learning*, 2009a.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, pages 11:19–60, 2010a.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 2010b.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *CVPR*, 2008a.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Proc. Neural Information Processing Systems*, 2008b.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *International Conference on Computer Vision*, 2009b.

J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE TIP*, 2008c.

J. Mairal, G. Sapiro, , and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modeling and Simulation*, 7:214 – 241, 2008d.

M.J.Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

R. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, pages 9(2):249–265, 2000.

B.A. Olshausen and D.J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 1997.

P. Orbanz and J. M. Buhmann. Nonparametric bayesian image segmentation. *International Journal of Computer Vision,*, page 77:25C45, 2008.

J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *Proc. International Conference on Machine Learning*, 2009a.

J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *ICML*, 2009b.

John Paisley and Lawrence Carin. A stick-breaking construction of the beta process. Technical report, Duke University, ee.duke.edu/∼jwp4/StickBP.pdf, 2009c.

O. Papaspiliopoulos and G. Roberts. Retrospective markov chain monte carlo methods for dirichlet process hierarchical models. *Biometrika*, 2008.

L. Ren, L. Carin, and D. Dunson. The dynamic hierarchical dirichlet process. In *International Conference on Machine Learning*, 2008.

L. Ren, L. Du, D. Dunson, and L. Carin. The logistic stick breaking process. *J. Machine Learning Research*, preprint, 2011.

L. Ren, D. B. Dunson, S. Lindroth, and L. Carin. Dynamic nonpara- metric bayesian models for analysis of music. *J. American Statistical Association*, 2009.

A. Rodriguez and D. Dunson. Nonparametric bayesian models through probit stick-breaking processes. *Bayesian Analysis*, pages 145–178, 2011.

R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. International Conference on Machine Learning*, pages 880–887, 2008.

M.A. Sato. Online model selection based on the variational Bayes. *Neural Computation*, 2001.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4: 639–650, 1994a.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4, 1994b.

Y. W. Teh, D. Görür, and Z. Ghahramani. Stick-breaking construction for the Indian buffet process. In *AISTATS*, volume 11, 2007.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *J. Am. Stat. Ass.*, 101, 2004a.

Y. W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101, 2004b.

J. B. Tenenbaum, V. Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319C2323, 2000.

R. Thibaux and M. Jordan. Hierarchical beta processes and the indian buffet process. In *AISTATS*, 2007a.

R. Thibaux and M. I. Jordan. Hierarchical beta processes and the Indian buffet process. In *Proc. International Conference on Artificial Intelligence and Statistics*, 2007b.

R. Thibaux and M.I. Jordan. Hierarchical beta processes and the Indian buffet process. In *AISTATS*, 2007c.

M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, June 2001.

C. Wang and D. Blei. Variational infernece for the nested chinese restaurant process. In *NIPS*, 2009.

C. Wang, D. Blei, and L. Fei-Fei. Simultaneous image classification and annotation. In *CVPR*, 2009.

C. Wang, J. Paisley, and M.D. Blei. Online variational inference for the hierarchical dirichlet process. In *AISTATS*, 2011.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

L. Wang and X. Wang. Hierarchical dirichlet process model for gene expression clustering. *EURASIP Journal on Bioinformatics and Systems Biology*, page 2013:5, 2013.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, and L. Carin. Nonparametric bayesian dictionary learning for analysis of noisy and incomplete images. 2011a.

M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, and L. Carin. Nonparametric Bayesian dictionary learning for analysis of noisy and incomplete images. *IEEE TIP*, 2012.

M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-parametric Bayesian dictionary learning for sparse image representations. In *NIPS*, 2009.

M. Zhou, C. Wang, M. Chen, J.Paisley, D. Dunson, and L. Carin. Nonparametric bayesian matrix completion. In *IEEE Sensor Array and Multichan- nel Signal Processing Workshop*, 2010.

M. Zhou, H. Yang, G. Sapiro, D. Dunson, and L. Carin. Dependent hierarchical beta process for image interpolation and denoising. In *Proc. Artificial Intelligence and Statistics (AISTATS)*, 2011b.

M. Zhou, H. Yang, G. Sapiro, D. Dunson, and L. Carin. Dependent hierarchical beta process for image interpolation and denoising. In *AISTATS*, 2011c.

Biography

Lingbo Li was born on November 6, 1986 in Xi'an, China. She received her B.S. degree in Electric Engineeing in Xidian University, Xi'an in June 2008, China. Beginning August 2008, she began to study in the department of Electrical and Computer Engineering at Duke University, where she received the M.S. degree in December 2010. She is expected to receive her PhD degree in Electrical and Computer Engineering at Duke University in May 2014. Her current research interests include Bayesian statistics and machine learning.

Publications

1. Lingbo Li, XianXing Zhang, Mingyuan Zhou, Lawrence Carin, "Nested Dictionary Learning for Hierarchical Organization of Imagery and Text", to appear in *Proc. Uncertainty in Artificial Intelligence (UAI2012)*, Catalina Island, Aug. 2012.

2. Mingyuan Zhou, Lingbo Li, David Dunson, Lawrence Carin, "Lognormal and Gamma Mixed Negative Binomial Regression ", to appear in *Proc. International Conference on Machine Learning (ICML2012)*, Edinburgh, Scotland, Jun. 2012.

3. Lingbo Li, Jorge Silva, Mingyuan Zhou and Lawrence Carin, "Online Bayesian Dictionary Learning for Large Datasets," to appear in *International Conference on Acoustics, Speech and Signal Processing (ICASSP2012)*, Kyoto, Japan, Mar. 2012.

4. Mingyuan Zhou, Haojun Chen, John Paisley, Lu Ren, Lingbo Li, Zhengming Xing, David Dunson, Guillermo Sapiro and Lawrence Carin, "Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images," *IEEE Trans. Image Processing*, Vol. 21, pp. 130-144, Jan. 2012.

5. Lingbo Li, Mingyuan Zhou, Guillermo Sapiro and Lawrence Carin, "On the Integration of Topic Modeling and Dictionary Learning," in *Proc. International Conference on Machine Learning (ICML2011)*, Bellevue, WA, Jun. 2011.

6. Lingbo Li, Mingyuan Zhou, Eric Wang and Lawrence Carin, "Joint Dictionary Learning and Topic Modeling for Image Clustering," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP2011)*, Prague, Czech Republic, May 2011.